Développement d'architectures HW/SW tolérantes aux fautes et auto-calibrantes pour les technologies Intégrées 3D

Vladimir Pasca

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité: Nanoélectronique et Nanotechnologies

Arrêté ministériel: 7 août 2006

Numéro ISBN 978-2-11-129172-0

Présentée par

Vladimir PASCA

Thèse dirigée par Mme. Lorena ANGHEL et
Codirigée par M. Mounir BENABDENBI

Préparée au sein du **Laboratoire TIMA**
Dans **l'École Doctorale d'Electronique, Electrotechnique, Automatique et Traitement du Signal**

Développement d'Architectures HW / SW Tolérantes aux Fautes et Auto-calibrantes pour les Technologies Intégrées 3D

Thèse soutenue publiquement le **11 Janvier 2013**, devant le jury composé de:

M. Ian O'CONNOR
Professeur, Ecole Centrale Lyon (Président)
M. Patrick GIRARD
Directeur de Recherche CNRS (Rapporteur)
M. Andreas STEININGER
Professeur, TU Wien (Rapporteur)
M. Pascal VIVET
Chercheur CEA-Leti (Examinateur)
Mme. Lorena ANGHEL
Professeur, INP Grenoble (Directeur)
M. Mounir BENABDENBI
Maître de Conférence, INP Grenoble (Co-encadrant)

Acknowledgements

It is always difficult to express your gratitude and acknowledgement to many great people that have provided me valuable advices during my PhD. My first thoughts go to my supervisor Professor Lorena Anghel, whose guidance from the first internships in 2006 until the last days of my PhD helped me both at a professional and personal level. I would like to thank her for the excellent guidance, great suggestions, never-ending patience, support and encouragement. My deepest gratitude also goes to my co-supervisor, Professor Mounir Benabdenbi. His suggestions, patience and guidance made my work easier and helped it come to completion. My gratitude also goes to Dr. Michael Nicolaidis, the leader of the ARIS group in TIMA. I cannot even find the words to properly express the great impact his wonderful advices and guidance had on my work.

I thank Professor Ian O'Connor for accepting to be the president of the jury and for the numerous scientific exchanges we had during our collaboration in the 3DIM3 project. I would like to acknowledge the great work of Dr. Patrick Girard and Professor Andreas Steininger in reviewing my work. Their great researcher experience, patience and understanding provided great suggestions that helped improve the quality of this manuscript. I also thank Dr. Pascal Vivet for accepting to be part of the jury. I value his comments on my work and his deep understanding on difference challenges of 3D integration, which pointed me to very interesting directions.

I would also like to express my gratitude to M. Marcello Coppolla from ST Grenoble for allowing me to validate most of my work in a realistic environment using the Spidergon STNoC platform. I thank Dr. Riccardo Locatelli and M. Giuseppe Maruccia for their helpful support during the time I spent in the ST group.

My work would have been very difficult without the constant help and support from my colleagues in TIMA. First, I would like to acknowledge Claudia Rusu for her help and suggestions in the first part of my thesis. No work on NoCs would have been possible without the many fruitful discussions I had with Fabien Chaix, Saif-Ur Rheman and Gilles Bizot. I also thank Hai Yu, Diarga Fall, Wassim Mansour, Thieri Bonnoît, Yi Gang, and Panagiota Papavramidou for their help, support and suggestions. Finally, I would like to acknowledge Salma Bergaoui for her help and advices, and for taking the awful task of correcting the French summary of my thesis.

Finally, I thank my family and fiancée for their support during these years away from them. None of this would have been possible without their invaluable advices.

Table of Contents

List of Figures

Glossary

TSV	*Through-Silicon-Vias*
VDSM / UDSM	*Very-/Ultra-Deep Submicron*
GSI	*Giga-Scale Integration*
TSI	*Tera-Scale Integration*
KGD	*Known Good Die*
NoC	*Network-on-Chip*
SoC	*System-on-Chip*
MPSoC	*Multi-Processor System-on-Chip*
MP2SoC	*Massively-Parallel Multi-Processor System-on-Chip*
TAM	*Test Access Mechanism*
BIST	*Built-In Self-Test*
BISR	*Built-In Self-Repair*
ECC	*Error Correction Code*
TMR	*Triple Modular Redundancy*
FEC	*Forward Error-Correction*
D2D	*Die-to-Die*
D2W	*Die-to-Wafer*
W2W	*Wafer-to-Wafer*

INTRODUCTION

For more than five decades, electrical engineers, computer and material scientist have built integrated circuits with increasing complexity and capabilities. In the early days of silicon semiconductor industry, it was predicted that the integration density (i.e. number of transistors per chip) would double every 18 months (*Moore's Law*) [Moo65]. Nowadays, all system functionalities (i.e. processing, memory, I/O, analog) are integrated on a single chip (or package) with more than one billion transistors / chip (i.e. Giga-Scale Integration GSI). The 10^{12} transistors / chip (i.e. Tera-Scale Integration TSI) milestone should be reached by the end of this decade. In the System-on-Chip (SoC) integration paradigm, hardware and software designers work together to offer systems with high computational power that satisfy the performance, energy-efficiency and reliability requirements.

However, improvements of computing performance and energy-efficiency by technological scaling lead to high development cost. Multi-processing became a powerful strategy to increase system performance by leveraging data-level and instruction-level parallelism. User applications are divided in tasks that run in parallel on one or more processing elements (i.e. Multi-/Many-Processor System-on-Chip MPSoC). Figure I-1 represents the ITRS trends in many-core system complexity.

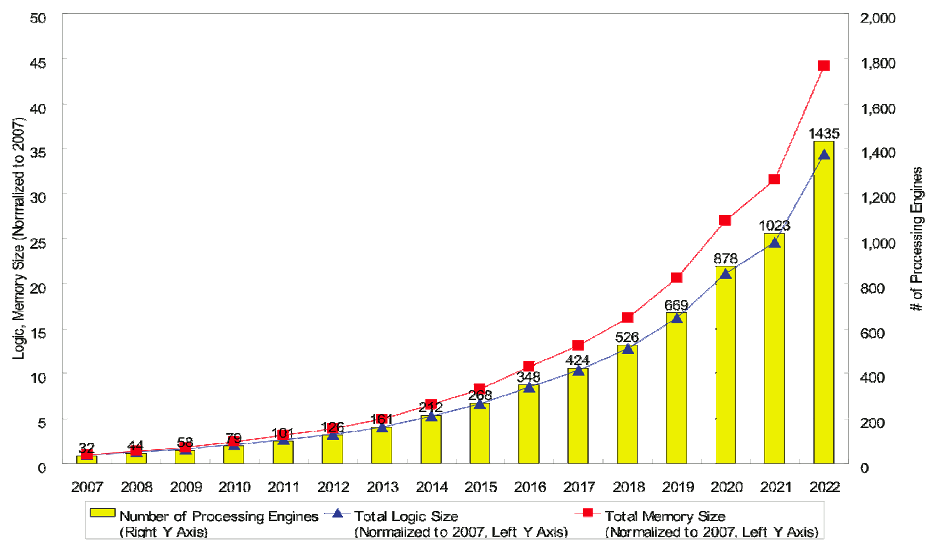


Figure I-1 Many-core system complexity trends [ITRS07]

The logic and memory size increases such that today chips comprise ~*5*× more logic elements and memory arrays than in 2007. By *2020*, chips are expected to have up to 20× more logic and 27× more memory components. The increase of logic size is mainly due to the increasing number of processing elements. As shown above, today we have just entered the hundreds-cores era, but within a decade chips are expected to have more than one thousand cores (many-core SoCs).

Unfortunately, the increased complexity of chips cannot be efficiently managed by traditional design paradigms. Technological scaling raises a serious issue that was overlooked in older technology nodes: *wires do not scale the same way as transistors*. In the last decade this problem gained a lot of attention since, in very-deep sub-micron (*VDSM)* technologies, wire delays begin to dominate the delays of logic gates. Despite well-known reliability issues (i.e. electro-migration, diffusion in silicon), the high conductivity of copper made it the suitable candidate to replace aluminum in chip wiring since 1998. Developments in copper interconnect technologies and *low-K* dielectrics reduced signal propagation delays and improved signal integrity. However, the delays of copper wires increase in advanced technology nodes while the gate delays decrease. For aluminum, the breakeven point was for the *250 nm* technology node (i.e. around year *2000*), while for copper it was at the *150 nm* technology node (i.e. around year *2005*). In Figure I-2, the wire and gate delay trends are represented for different technology nodes.

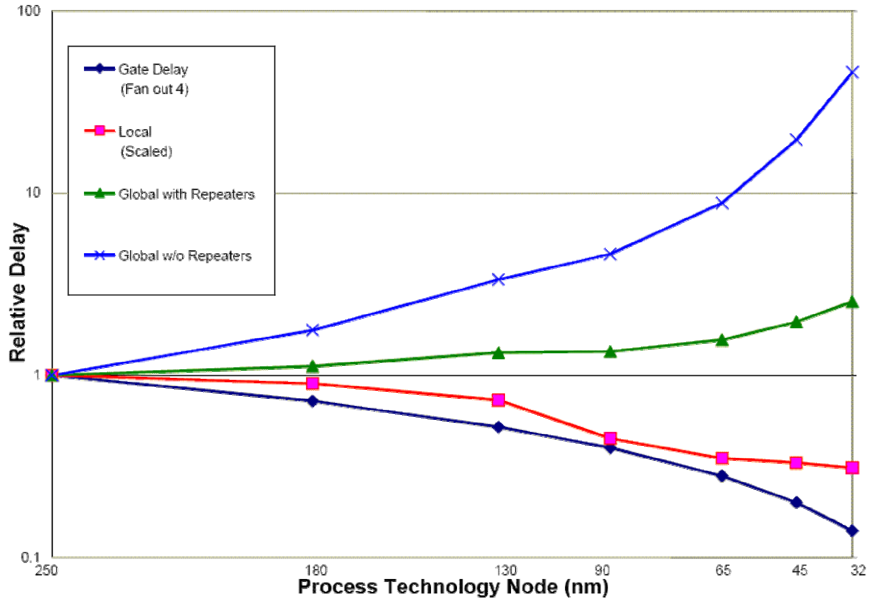


Figure I-2 Trends of interconnect and transistor delays for sub-micron technologies [ITRS05]

The trends above show that, relative to *250 nm* technology node, the gate and local wire (i.e. first metal layers M_1-M_3) delays decrease from generation to generation, while the global interconnect (i.e. top-most metal layers) relative delay increases. In the *32 nm* technology node, gates are up to 5× faster than in *250* nm technologies, while signal propagation on global wires is up to 50× slower. The delay challenge of global wires can be partially alleviated by inserting repeaters. In this case, the relative delays of global wires are 5× larger. However, the delay gain is paid in extra area and power for the repeaters. In [ML04], it was shown that the dynamic power dissipated on interconnects accounts for *~50%* of the total chip power consumption and about *90%* of this power is dissipated on global wires. For advanced technology nodes (i.e. below *32 nm*) that study also predicted that the dissipated on wires will account for *65%* of the total chip power. Therefore, interconnects are the performance bottleneck in GSI / TSI technologies.

Technology scaling poses another challenge: traditional bus-like interconnect fabrics are not scalable for SoCs comprising hundreds or thousands of Intellectual Property (IP) blocks. Communication contention due to arbitration reduces the overall system performance. In embedded MPSoCs, a widely accepted solution to this problem is the packet-switched interconnection fabric: the *Network-on-Chip (NoC)*. Similarly to the ISO/OSI model in computer networks, NoCs also have abstraction layers, which are shown in Figure I-3.

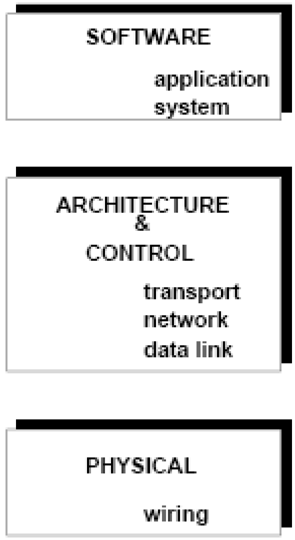


Figure I-3 Abstraction layers of NoC-based MPSoCs [BdM02]

In most communication-centric systems, on-chip communication is performed by load / store transactions (i.e. transaction-based communication), using a specific protocols (e.g. AMBA-AXI, VCI/OCP). Initiator (or master) nodes (e.g. CPU), begin transactions by making load / store requests to targets (or slaves) modules (e.g. memory). In software, application-level read / write operations to memory are translated into system-level load / store transactions. At transport-level, these transactions are split in request / response messages that are packetized and forwarded to the network. Packets are processed at network-level and they are transmitted from source to destination in a pipelined way. At data link-level, routers exchange flow control units (flits) over the physical communication environment *PHY*.

The interconnect problems can be solved at technological level using 3D integration. In 3D chips, silicon layers are stacked and connected by shorter vertical wires called Through-Silicon-Vias (TSVs). The power and timing improvements promises of stacked 3D integration are mainly due to the reduced average interconnect length. At system-level, the joint use of 3D integration and NoC-centric design paradigms open the way to novel energy-efficient architectures with significant performance improvements.

Despite recent technological progress, TSV-based 3D chips are not considered mature enough for large-scale production. Interconnect test, manufacturing costs, yield and reliability, thermal management and heat removal are among the greatest challenges of this technology. In 3D MPSoCs using NoCs as system-level interconnect fabric, testing and ensuring high reliability / yield becomes increasingly difficult. Intra-/inter-die communication reliability is paramount for communication-centric MPSoCs. Faults that occur during network traversal may lead to system failures. Hence, efficient interconnect test strategies and error resilience techniques are no longer a *nice-to-have*, but a *must-have* feature. In this thesis, three main issues of NoC-centric 3D MPSoCs are addressed: interconnect TSV test, yield and reliability.

Interconnect Built-In Self-Test (*IBIST*) strategies have proven to be efficient in testing on-chip and board-level interconnects for structural and functional faults. If inter-die synchronization can be guaranteed in stacked 3D chips, then similar *IBIST* methodologies can also be used for TSV tests. Although existing test pattern generators can be used for TSV tests, they have poor fault coverage (e.g. only structural faults like opens and shorts are detected) or they are too conservative, leading to high test costs. Using the victim-aggressor scenario, where signal transitions on victim wires are affected by signal transitions on aggressor wires, the *K^{th}-Aggressor-Fault* (KAF) model is proposed. For KAF-based TSV tests, depending on their distance to the victim TSV, neighboring aggressors are organized in classes called *aggressor orders*. The novelty of this approach is that, depending on the TSV technology, higher order aggressors may not be considered for tests. The major advantages of this strategy are simpler test circuitry and fewer test patterns, which translates into shorter test duration.

Error resilience of 3D NoCs against TSV permanent and transient faults can be efficiently improved by different single-/multi-layer strategies. In multi-layer schemes, the task of fault mitigation is shared across several abstraction layers (e.g. data link and network). In 3D NoCs, error resilience can be ensured at data link and / or network levels, on NoC components (i.e. links and routers). The main contributions of this thesis on error resilient strategies are summarized in the following.

In modern chips, transient faults are responsible for more than *80%* of system failures [BdM02]. In 3D NoCs, transients can affect both wires (i.e. intra-die wires and TSVs) and different router components (e.g. buffers, arbitration logic). To leverage transient errors on NoC links, correct data transmission is ensured by implementing retransmission-/correction-based error control schemes on intra-/inter-die links. Reliable transmission through the entire network (i.e. links and routers) is ensured by network-level error resilience schemes on individual flits. In the multi-layer framework, link-level and network-level protection are jointly used in order to ensure reliable communication. Flits are individually encoded at the source, and correction stages are inserted on links along the path such that multiple errors do not cumulate.

Permanent TSV faults due to manufacturing defects have a dramatic impact on 3D chip yield. The proposed data link-level solutions to repair TSV failures in 3D NoCs are: *Spare-and-Replace* (*TSV-SnR*) and *Configurable fault-tolerant Serialization* (*CSL*). At network-level, TSV permanent faults are mitigated by instructing routers to forward packets around faulty components (i.e. inter-die links). An important feature of the proposed TSV-fault tolerant routing algorithm (*TSV-FTR*) is that it does not use virtual channels to ensure deadlock freedom, a critical property for NoCs.

In aggressive 3D technologies, chips are expected to have *thousands* and *tens of thousands* TSVs. For such complex systems, in-field TSV failures may cause serious reliability issues. Therefore, a Built-In Self-Repair strategy based on spare TSVs and adaptive serialization (*IBIRAS*) has been developed. The main benefit of this approach is that no external intervention is required, as the repair signals are determined on-chip using the interconnect test diagnosis vector.

Although solutions to the TSV reliability / yield issues exist, it is difficult to determine which of these strategies is better suited for a given 3D NoC. Choosing the best error resilience configuration for a 3D NoC architecture, which must satisfy yield and reliability targets for a given failure rate, is very difficult. *One-fits-all* solutions do not exist, as each technique has its own advantages and limitations with respect to costs (i.e. area, power, TSV count) and impact on system performance. The area and power budget could be very limited and, depending on system requirements, the performance penalty due to error mitigation should be very low.

Moreover, the possibility to jointly use error resilience at different abstraction layers (i.e. data link and network) makes new costs-performance trade-offs possible. To address this challenge, an *error resilience exploration (ERX)* tool for 3D NoCs is proposed in this thesis. Hence, given a 3D NoC configuration, different error-resilient configurations (i.e. single-layer and multi-layer) are implemented and evaluated (i.e. area, power and network latency) such that designers can choose the optimal solution.

In the remaining of this manuscript, all above contributions are detailed as follows. Chapter 2 presents a state-of-the-art on 3D integration technologies and their implications on system architectures. Different 3D integration strategies and challenges, and (3D) NoC-related design, test and reliability issues with existing solutions are discussed. In Chapter 3, the novel KAF model used for TSV self-tests in NoC inter-die links is presented. The error resilience strategies and exploration tool for 3D NoCs is presented in Chapter 4. Data link and network-level fault tolerance solutions for transient and permanent faults are presented with different scenarios in which they can be jointly used in multi-layer error resilience schemes. Chapter 5 presents an assessment of the error resilience schemes in the context of 3D mesh NoCs. The objective of this study is to show the resilience capabilities of each solution and identify different trade-offs. The *error resilience exploration* concept with its application to 3D NoCs is presented in Chapter 6. Using the error resilience exploration tool (*ERX*), the impact of error resilience on system performance is assessed for a *64*-tiles 3D MPSoCs which is implemented in SystemC using the SoCLib cycle-accurate bit-accurate (CABA) library [SL]. Finally, Chapter 7 concludes this thesis and discusses further research directions.

Chapter Two

3D Integration and Networks-on-Chip

Stacked 3D integration is an emerging technology that promises heterogeneous system integration with reduced power consumption and increased performance. Layers of active silicon are stacked and vertically connected by Through-Silicon-Vias (TSVs). Today, TSV-based 3D integration is not yet fully mature and new design and test concepts are being developed. While novel techniques are used to improve the 3D chip compound yield and reliability, several other techniques have been adapted from well-established fields. In this chapter, TSV-based 3D integration fundamental concepts and challenges are presented. As the main focus of this thesis are the test and reliability / yield issues of the 3D system interconnect fabric (i.e. Network-on-Chip), different solutions to the above mentioned challenges are presented in the context of (3D) NoCs.

2.1 *3D Integration using TSVs*

The System-on-Chip (*SoC*) paradigm primarily addresses the complexity of off-chip interconnects by integrating all system components in a single package. From Multi-chip-Modules (*MCMs*) to Systems-in-Package (*SiPs*) and Systems-on-Package (*SoPs*), the number of components / cm^2 has steadily increased in the last decades. Although systems become more complex and offer more functionality, integrating them in a single package becomes a great challenge. The key technology for more system-level integration is 3D stacking using Through-Silicon-Vias (*TSV*s).

The benefits of TSV-based 3D integration are mainly achieved by stacking active silicon layers. These layers are connected using short (i.e. *tens* of *μm*) inter-die wires called Through-Silicon-Vias (TSVs), as represented in Figure II-1 (a). In 3D Systems-on-Chip (3D SoC), performance improvements and power savings are achieved by replacing long global interconnects of 2D SoCs with shorter TSVs. Compared to 2D chips, the average interconnect length is reduced by a factor proportional to the square root of the number of layers in the stack [PF09]. In Figure II-1 (b), the connection between blocks *A* and *C* of a 2D SoC is replaced by TSVs, significantly reducing the wire length.

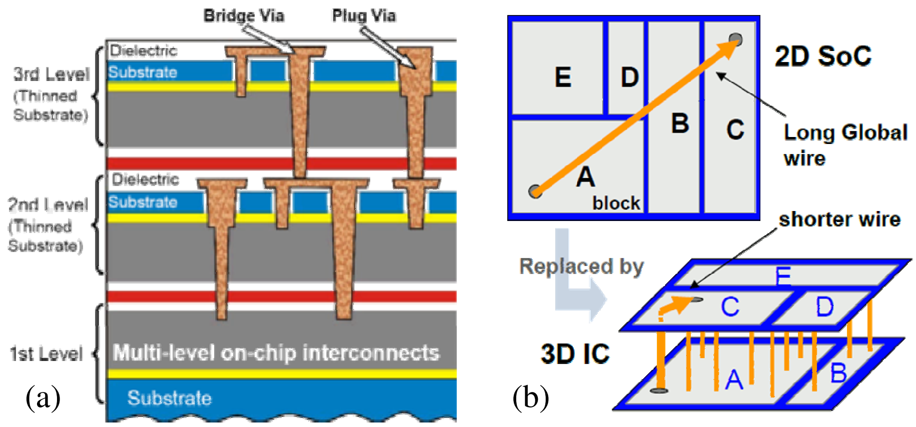


Figure II-1 TSV-based stacked 3D integration (a) [Lu03] reduces the average global interconnect length (b)

Unlike traditional chips, each layer of the 3D stack can be implemented in different technologies. The advantages of TSV-based 3D integration are not limited to integration heterogeneity. A power reduction of *40%* was reported in [Pat11] for a stacked 3D chip. In the same paper, the chip footprint of 3D ICs is also reduced and more capabilities can be integrated in a single package (i.e. up to *4×* density increase). Moreover, the flexibility of 3D system architectures enables an up to *400%* performance improvement.

The manufacturing processes of 3D chips are more complex than in traditional CMOS. There are many different processing technologies to choose from. Each process has its advantages and disadvantages in terms of compatibility with existing CMOS manufacturing processes, manufacturing throughput and reliability. The main steps in manufacturing 3D chips are wafer thinning, TSV formation and bonding. The order in which these processes are performed can be different, leading to different trade-offs. In Figure II-2, the main manufacturing processes with respect to TSV formation are presented.

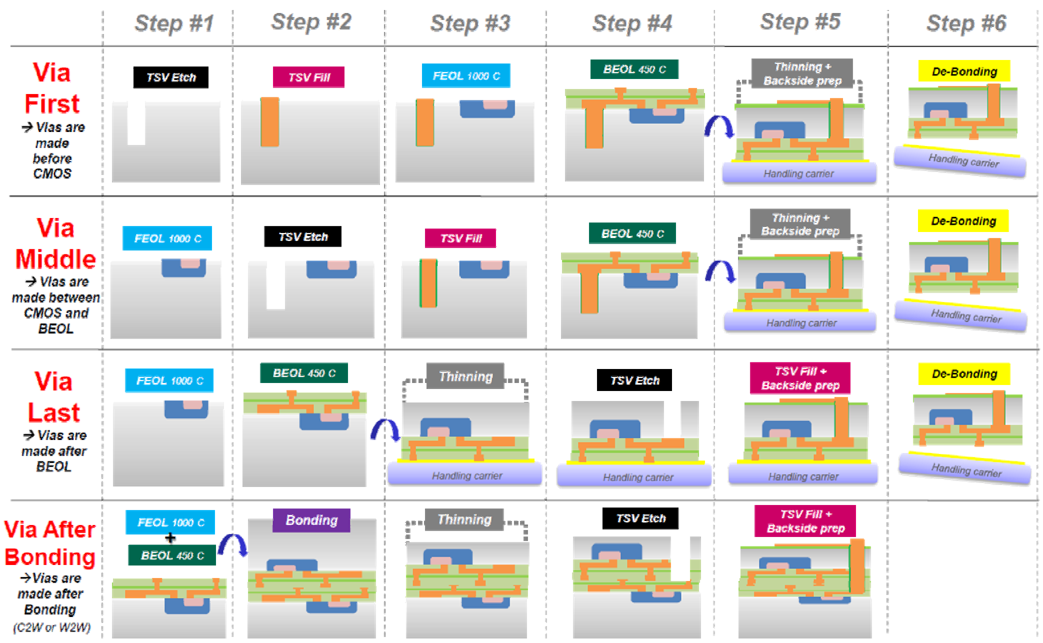


Figure II-2 Summary of 3D chip manufacturing steps (source: YOLE Development)

In the *via-first* approach, the TSVs are manufactured first by etching the bulk silicon and depositing the TSV metal (i.e. copper or tungsten). After the lithographic processes for transistors manufacturing and metal layers deposition, the wafer is flipped and, while bonded to a carrier wafer, the bulk silicon is thinned down to tens of μm (i.e. until the TSV tips are visible). Before stacking the wafer, backside preparation processes and de-bonding from the handling carrier are performed. In the *via-middle* approach, TSVs are formed in intermediate steps between transistor manufacturing and metal layers deposition. In the *via-last* approach, after the lithographic processes for transistors and metal layer, the wafer is flipped. Then, the wafer is thinned while bonded to a carrier wafer and TSVs are formed. After backside preparation processes, the carrier wafer is removed and the thinned wafer is ready for bonding. In the *via-after-bonding* approach, TSVs are manufactured after the wafers bonding and thinning processes.

Depending on the die/wafer orientation, bonding can be made *face-to-face*, *back-to-back* and *face-to-back*. It is also possible to have *wafer-to-wafer (w2w)*, *die-to-wafer (d2w)* and *die-to-die (d2d)* bonding. In the *w2w* approach, entire wafers are bonded. The advantage of this approach is high manufacturing throughput. Unfortunately, this strategy can result in poor compound yield, as faulty dies are likely to be stacked on functional ones. Another limitation of wafer-level processing is that dies must have the identical sizes, while in the *d2w* strategy dies with different sizes can be stacked. The *d2w* process also has the advantage of higher yield, but the manufacturing throughput is reduced, as individual dies are tested before bonding (i.e. *Known-Good-Die KGD*). The *d2d* approach has the potential of providing maximum yield, as only dies that pass pre-bond tests are stacked. However, similar to the *d2w* case, the throughput is significantly reduced and test / manufacturing costs are higher.

TSV-based stacked 3D integration is used in different applications such as CMOS Image Sensors (CIS) [HJN08], and memories [KCH10], *Memory-on-Logic* (*MoL*) [Loh08, HAG10, VG11], and *Logic-on-Logic* (*LoL*) [FDG12]. *MoL* systems comprise a single layer of interconnected processing cores (ASIC) which exchange data with memory layers. In *LoL* MPSoCs, two or more stacked dies of processing cores exchange messages. In this case, the global interconnect fabric, which connects these cores, spans across two or more layers, ensuring both intra-die and inter-die communication.

3D integration is an emerging and rapidly growing technology. Despite its promises, several challenges must be addressed before 3D stacking becomes main-stream. In the following section, the major challenges of TSV-based 3D integration and potential solutions are presented.

2.2 *Challenges of TSV-based 3D Integrated Systems*

Testing, reliability and yield, thermal management and costs remain the major challenges of TSV-based 3D integration. In recent years, these issues have been addressed by the scientific community using novel solutions or re-implementing existing ones. In this section, a review of major contribution in solving these challenges is presented.

2.2.1 ***Testing TSV-based 3D ICs***

Among the most important challenges is how to efficiently test 3D system architectures. A common strategy is to test each die before bonding (i.e. *Known Good Die* bonding) and then perform final chip tests. It has been shown that this strategy pays-offs only when the die yield is not very high [Mar10]. For a 3D chip comprising many dies, it is also possible to perform partial stack dies testing. In other words, each time a die is stacked, the existing partial stack is tested. In Figure II-3, a 3D chip test flow with intermediate pre-bond and post-bond test steps is represented [MZ09].

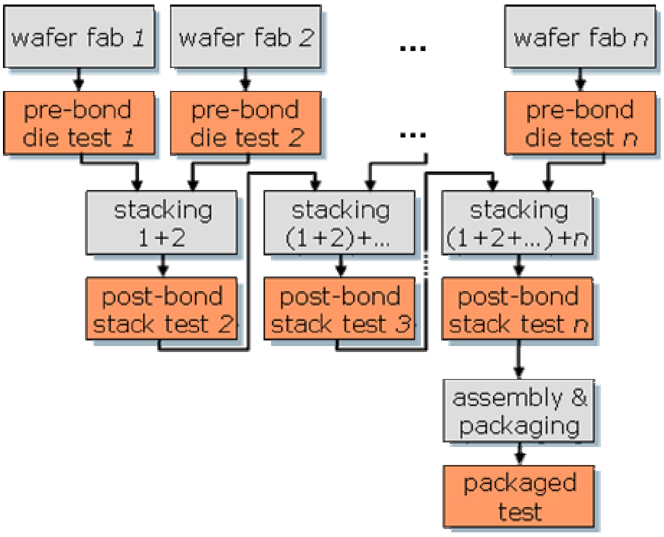


Figure II-3 Stacked 3D chip test flow [MZ09]

The manufacturing throughput of the *partial-stack test* strategy is very low, as tests are performed both *pre*- and *post*-bonding. In order to improve manufacturing throughput, it is possible to remove intermediate test steps. A challenge in *pre*-bond tests is that it requires access to the die using dedicated access pads. However, because these tests are often performed on thinned dies/wafers, reliability problems arise, as test pins could damage the die/wafer [MZ09].

Design-for-Test (DfT) circuitry is required to ensure full controllability and observability of 3D system components (i.e. TSVs, cores, dies). In order to ensure efficient testing, there are several requirements that the Test Access Mechanism (3D TAM) should satisfy. The 3D DfT architecture must be scalable with respect to the number of dies. In order to have higher production testing, a parallel access mechanism, which provides a trade-off between implementation costs and test access bandwidth, should be considered [MZ09, Mar10, MCK12]. Another possible requirement of 3D test is modularity: cores and TSVs are tested as separate units. If this requirement is fulfilled then it is possible to optimize testing for specific fault models, enable test flow optimization optimize the number of test elevators (i.e. TSVs used for test data transmission in upper layers) [NGC10] or reduce test duration [NCG11]. In Figure II-4, an example of state-of-the-art 3D TAM based on the IEEE 1149.1 and IEEE 1500 standards is given [MCK12].

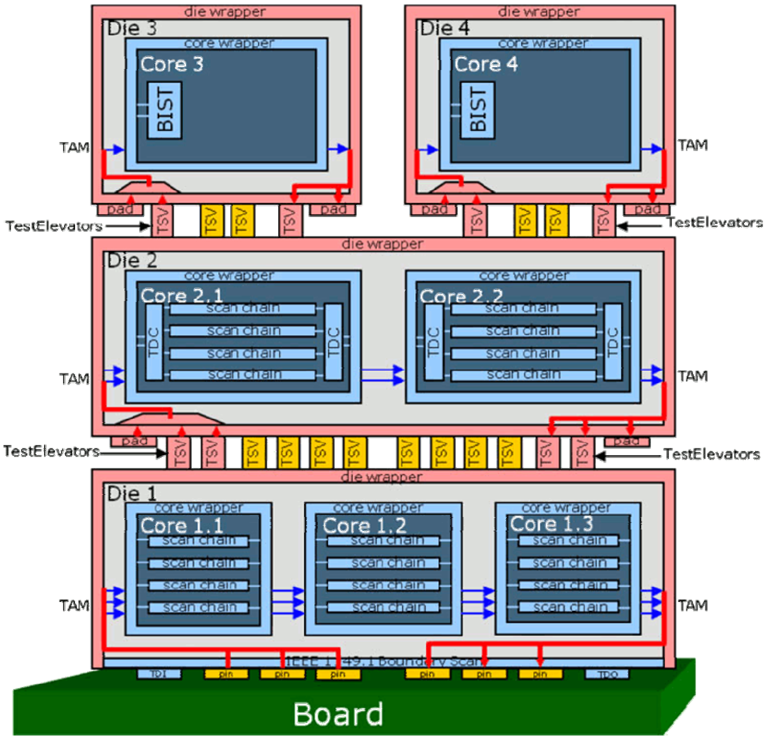


Figure II-4 Example of a Simple 3D Test Acess Mechanism [MCK12]

In this 3D DfT architecture, die-level test wrappers are considered for each die in the stack. The main features of the standardizable die wrapper are serial and parallel interfaces for wrapper instructions / low-bandwidth tests and high-bandwidth test data; *TestTurns* that feed data back to the bottom die; *Test Elevators* that propagate test data up and down the stack; optional test pads for pre-bond testing; and an optional inclusion / exclusion mechanisms for embedded IPs.

The most important feature of the 3D TAM is the possibility of inter-die wire testing. Interconnect tests can be performed similarly to boundary scan (IEEE 1149.1) and built-in self-test strategies in on-chip and board-level wires. In the following, interconnect failure modes and strategies to sensitize these faults are presented.

2.2.2 Testing Through-Silicon-Vias

The TSV failure modes are not fundamentally different from on-chip and board-level interconnects. In all cases, interconnect manufacturing and aging/wear-out defects are modeled using the basic faults: *open*, *short*, *stuck-at*, and *delay* faults. In the *open fault model*, a wire is assumed broken and its terminals are electrically disconnected. In the *short fault model*, two ore more wires are electrically connected when they should not be. Depending on the technology, either *AND* or *OR* logic gates can be used to obtain the resulting value of the wires. For *stuck-at* faults, the signal sent on a wire is stuck at a *'0'* or *'1'* value, independently of the values sent on the wire. In the case of *delay faults*, the signal eventually assumes the correct value, but more slowly (or rarely, more quickly) than normal. Possible causes of delay faults are process variation, crosstalk, etc.

Sensitizing these faults requires dedicated test circuitry that ensures both interconnect controllability and observability. In the following it is shown how the boundary scan and interconnect built-in self-test strategies, which have been used for on-chip interconnect tests, are used for TSV tests.

2.2.2.1 *Boundary Scan*

The main purpose of the commonly used IEEE 1149.1 (Boundary Scan) standard is Printed Circuit Board (PCB) wire testing. This standard can be adapted for 3D ICs such that TSV interconnect test can be performed in a similar way. The major advantages of *Boundary Scan* are a reduced number of test pins and a relatively small number of test patterns necessary to sensitize structural faults (e.g. open, shorts). To illustrate how TSVs can be tested, let us consider the two stacked dies such that the lower die is connected with the upper die through *N* TSVs. The simplified *boundary-scan* architecture is presented in Figure II-5.

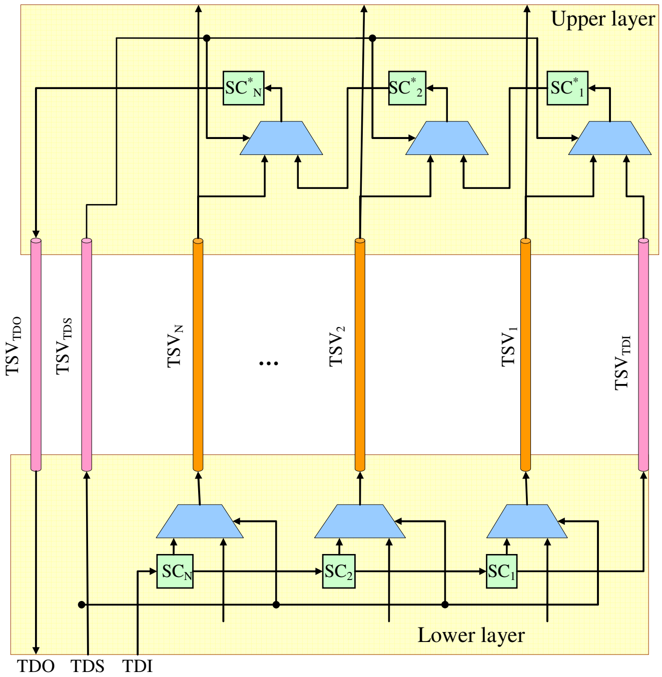


Figure II-5 Boundary Scan Test for regular TSVs TSV_1-TSV_N

The *N*-bits test vector T_1-T_N is serially shifted in the chip through the *TDI* port and transported to the lower scan cells SC_1 to SC_N. Once the test vector is loaded in these cells, the control signal *TDS* is set and the scan cells contents is transmitted through the *N* TSVs and loaded in the SC_1^* to SC_N^* scan cells from the upper layer. Then, under the control of the external tester, the response vector T_1*-T_N* (i.e. contents of SC_1^* to SC_N^*) is shifted out on the *TDO* port. In order to reduce test times, while the response vector is shifted out, the next test vector may be loaded in SC_1 to SC_N. After the test phase, the external tester compares the test and response vectors in order to identify faulty TSVs. Note that the TSV_{TDI}, TSV_{TDO} and TSV_{TDS} vertical connections must be functional in order to guarantee test correctness.

Test vectors are generated off-chip using different algorithms based on counting sequences [Kau74], modified counting sequences [GM82], true/complement sequences [Wag87], marching ones, etc. Since tests are performed at the lower frequencies of external testers, only structural faults can be detected.

2.2.2.2 *Interconnect Built-In Self-Test*

TSV interconnect delay faults testing can be performed by adopting at-speed test strategies similar to those developed for on-chip and board-level interconnects. However, it is necessary to have good inter-die synchronization, for such implementations to be possible.

Interconnect Built-In Self-Test (*IBIST*) strategies are used for testing global on-chip and board-level interconnects. Implementing *IBIST* strategies in TSV-based 3D systems comes with some advantages: fewer test pins / test elevators and simplified external testers, as test vectors (or test stimuli) are generated on-chip. Because tests are performed at nominal clock rates (i.e. at-speed tests), delay faults (including delay faults due to crosstalk) can also be sensitized. Let us consider a TSV bundle with *N* wires that connect two stacked dies. In Figure II-6, a simplified *IBIST* test architecture adapted for 3D interconnects is represented.

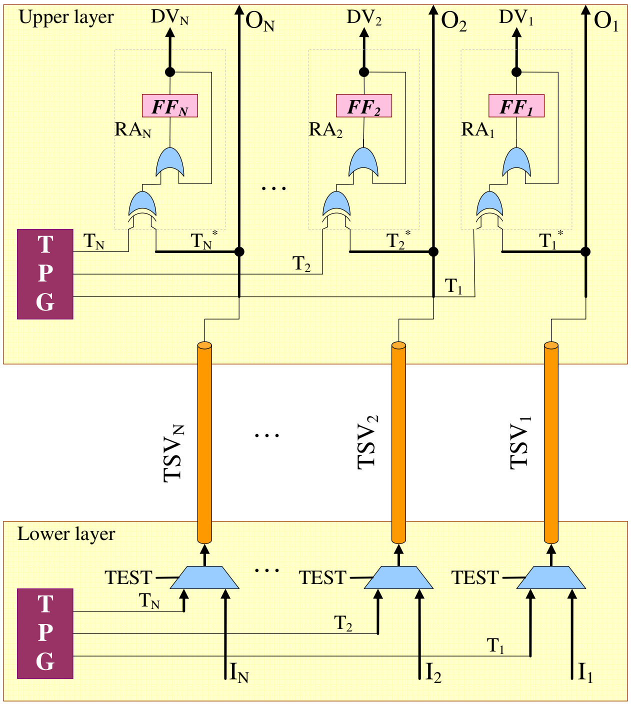


Figure II-6 Interconnect Built-In Self-Test for *N* regular TSVs

During the test phase (i.e. *TEST='1'*), *N* bits test vectors T_1-T_N are generated by Test Pattern Generators (*TPGs*) in the lower die and transmitted on TSVs. In the upper layer, the received vector T_1^*-T_N^* is compared to the original transmitted vector locally generated by a second *TPG* identical to the one in the lower layer. The *response analysis* cells (*RA*) compare the response and test vectors, and iteratively build the diagnosis vector DV_1-DV_N. In each response analysis cell RA_i, a *XOR* gate compares the received test stimulus T_i^* with the expected one T_i. Using an *OR* gate, any mismatch of these signals is stored in *RA's* internal flip-flop F_i. At the end of the test phase, the diagnosis vector which identifies the faulty TSVs is stored in *RA*'s internal flip-flop FF_1-FF_N.

The *IBIST* main component is the *TPG* block. Test patterns can be generated using different algorithms, depending on the targeted TSV fault models (e.g. open, short, delay). In the case of open and short faults, the *TPG* module generates test patterns using algorithms such as Counting Sequence, Modified Counting Sequence, or True/Complement Sequence. In order to detect delay faults, both *'0'-'1'* and *'1'-'0'* transitions have to be activated for every wire. The marching-'1'/'0' and rearranged True/Complement test sequences can be used for delay faults. Interconnect at-speed tests using Linear Feed-back Shift Registers (LFSRs) and Multiple-Input Shift Registers (MISRs) have been proposed for on-chip interconnect tests [SD02]. LFSR optimization techniques are also possible. For example, the authors of [PCZ01] have used Markov chains to generate realistic switching activity. The disadvantage of *LFSR*-based solutions are high aliasing probability (i.e. multiple faults impact the MISR's capability to detect them), long test sequence for detecting all faults, and relatively low fault coverage.

In [Jut04], at-speed tests for delay faults are performed using *Interleaved True/Complement Counting* (*ITCC*) sequences where complemented vectors are transmitted right after their true valued counterparts. *ITCC* sequences sensitize some delay faults due to crosstalk, as the complemented values '0'/'1' and '1'/'0' are transmitted on any two pair of wires. In [CDB99] a worst-case test strategy for delay faults due to crosstalk has been proposed for on-chip interconnects. This strategy can detect both structural / delay faults, but the test sequence is very long (i.e. for *N* wires a total of *8N* vectors are used, compared to the $2 \cdot \lceil log_2(N) \rceil$ vectors of *ITCC*) and the hardware implementation costs are non-negligible. Using the aggressor-victim scenario, the *Maximum Aggressor Fault (MAF)* model proposed in [CDB99] takes into account the crosstalk effects between a set of aggressor wires and a victim wire. Signal transitions on the victim wire are affected by the crosstalk noise induced by transitions on *all* other wires.

Few of these delay faults strategies have been implemented in a 3D integration setting. For example, LFSR-MISR techniques have been implemented for TSV self-test [HHH10]. In this test strategy, which is represented in Figure II-7, TSVs are assumed are distributed on regular arrays.

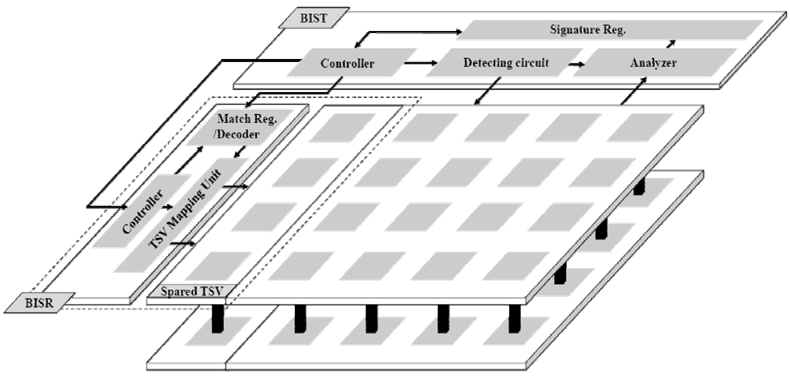


Figure II-7 Built-In Self-test and Repair for Through-Silicon-Vias [HHH10]

In order to reduce test durations, TSVs are tested one row at a time and a single spare is allocated for every row. The number of dedicated TSVs required by this self-test and self-repair strategy is high: for each TSV row, a single TSV is necessary to indicate in the lower die that the functional spare replaces a faulty regular TSV.

Another post-bond interconnect TSV *BIST* strategy was proposed in [HLC11]. In this scheme, it is also considered that TSVs are distributed on regular arrays. In Figure II-8, the BIST architecture for the TSVs of a die is represented. In this architecture, *outbound* TSVs are the TSVs that carry signals off the current die and *inbound* TSVs are those TSVs that carry signals from the neighboring die.

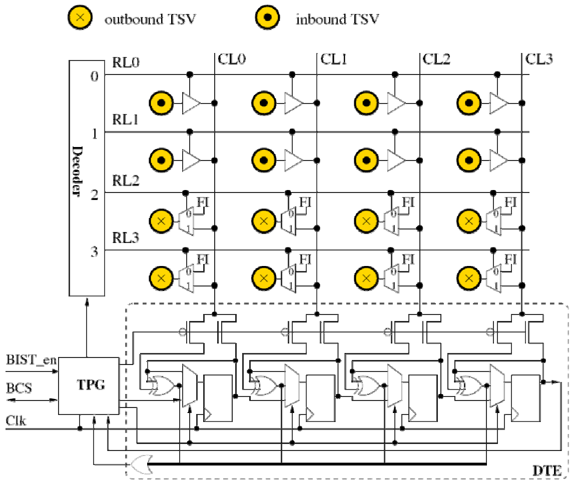


Figure II-8 Post-bond TSV BIST for bidirectional TSVs set as inboud and outbound for a given die [HLC11]

The *BIST* consists of a *row decoder*, a *test pattern generator* (*TPG*), and a *test data transportation and evaluation* (*DTE*) circuit. The BIST control signals (*BCS*) and enable signal (*BIST_en*) manage the *BIST* operation. In order to reduce test and diagnosis duration, TSVs are tested one row at a time. The decoder selects a row for test pattern application or test response evaluation. The TPG generates control signals and test patterns for the decoder and the *DTE*. The *DTE* consists of shift registers and a comparator such that it can perform the test pattern/response transportation using the shift register and the test response evaluation using the comparator. TSVs in the same column are connected to a column line (*CL*) through a multiplexer or a tri-state buffer. For an outbound TSV, a multiplexer is used to select a signal from the functional circuit (*FI*) or the CL. For an inbound TSV, a tri-state buffer is used to pass or block the signal to the test circuit. The benefit of this approach is that it requires less time and on-chip resources than IEEE 1500 test wrappers. However, $1+c+2\cdot r$ cycles are needed to test the $r\times c$ TSV array for a single test pattern.

Recent advances in testing 3D ICs and TSVs enable chip manufactures to adopt 3D integration. Moreover, the compatibility of 3D DfT with well-established IEEE standards such as IEEE 1149.1 and IEEE 1500 will facilitate the use of this technology in future designs. However, for products to be successful, testing is only part of the solution. The 3D chip reliability and yield challenges and potential solutions are discussed in the following section.

2.2.3 Reliability and Yield

The key to any successful product, including 3D MPSoCs, is whether it is reliable (i.e. there are very few failures during its expected lifetime) and the manufacturing yield is high enough such that very few units are discarded due to manufacturing defects. In order to ensure high reliability and yield, faults that may occur during manufacturing processes and system lifetime must be mitigated. In this section, major fault tolerant solutions proposed for 3D ICs are discussed. However, in order to understand how fault tolerant solutions are used, a brief classification of faults is presented first.

2.2.3.1 Faults classification

Depending on their nature, faults can be *permanent* or *temporary* (i.e. *transient* and *intermittent*). In general, a permanent fault always occurs when a particular set of conditions exists. In hardware, permanent faults model physical defects and they affect the circuit behavior always in the same way. They usually include the faults due to manufacturing process defects. During normal operation, a number of aging / wear-out mechanisms can occur in the long term. Such faults (e.g. electro-migration in wires) are initially revealed as intermittent faults until they finally provoke a permanent fault.

A *transient fault* disturbs the normal operation of an element in the system for a limited period of time. Outside this time period, the affected element operation is not influenced. Transients usually do not cause physical damage to the device, but the effects of a transient fault may last even after its duration ends. Transient faults in MPSoCs are mainly represented by the so-called *soft errors*. These are caused by interferences of any kind, but also by the impact of cosmic rays and radiation on silicon. Soft errors generated by radiation impact are mostly present in harsh environments with very high concentration of radiations or particles (e.g. space or nuclear power plants). However, with the advanced technology downscaling and the voltage and capability reduction, the circuit sensitivity to soft errors is significant even in normal environments [Nic05].

An *intermittent fault* is defined as a malfunction of a device or system that occurs periodically, either at regular or irregular intervals. Outside of these periods, the device or system functions normally. The cause of an intermittent fault is several contributing factors occurring simultaneously. Intermittent faults in interconnects typically cause burst errors.

Single or *multiple* faults can occur in the system. They can occur in one or several parts of the system: transmission lines and combinational logic wire values can be delayed or inverted, while register values and memory bits can be inversed.

Due to the increased complexity of 3D ICs, errors are more than a possibility. Moreover, intra-die and inter-die parametric variations and errors cumulate, leading to potentially low yield and reliability. In the following, different yield improvement strategies that mitigate permanent faults due to manufacturing defects are presented.

2.2.3.2 *Manufacturing defects in 3D ICs*

Pre-bond die tests help improving the chip yield, as only functional dies are stacked (i.e. KGD bonding). In [THV10], a compound yield improvement strategy for wafer-level processing is proposed. Using a *wafer repository*, wafers are bonded such that faulty dies are on the same position. Using this strategy the probability that packages contain faulty dies can be significantly reduced. In [Sin11], the symmetry of wafer has been used for matching faulty dies and improving the overall chip yield.

Another yield improvement strategy consists in allocating spare layers. Unfortunately, this strategy can be efficiently implemented only for 3D memories, which have highly regular architectures. For example, the authors of [TH11] use redundant layers and dedicated layer replacement circuitry that alleviates the issues of electrical fusing by using a series of address comparators.

In the *KGD* paradigm, TSV defects are the main cause of yield loss. Void formation, lamination due to thermally induced stress, height variation, and XY misalignment are major TSV failure mechanisms [PCF07, KXM09, LCD09].

Inter-die wire failures due to manufacturing are often mitigated using *spare-and-replace* strategies (i.e. *hardware redundancy*) where functional spare TSVs replace faulty regular TSVs. Hardware redundancy is proposed in [HHC10] for ASIC applications where TSVs are packed in blocks with one spare per block. Signals are routed on TSVs through a chain of MUXes. When faulty TSVs are detected, the signals transmitted on that TSV and on all subsequent TSVs in the chain are shifted by one position. The MUXes are controlled by arrays of e-fuse memories that are programmed using scan chains. For TSV failure rates (i.e. probability that a single TSV is faulty) up to *0.01%*, one spare is allocated for each TSV block such that the TSV yield is raised up to 99.99%. However, the fault recovery rates (i.e. the probability that the fault pattern can be repaired) drop to 90% and 95%, when there are 50 and 25 TSVs per block, respectively. Hence, for the same number of TSVs, higher yield can only be achieved by partitioning the TSVs in more groups and allocating more spares / chip.

For 3D DDR3 DRAM memories [KCH10], TSV connectivity check and repair is used to improve the yield (>98%) of stacked chips comprising four layers connected by *300* TSVs. In the proposed schemes, which are represented in Figure II-9, one additional spare is considered for every pair of signals (2:1) or two additional spares are added for every four signals (4:2).

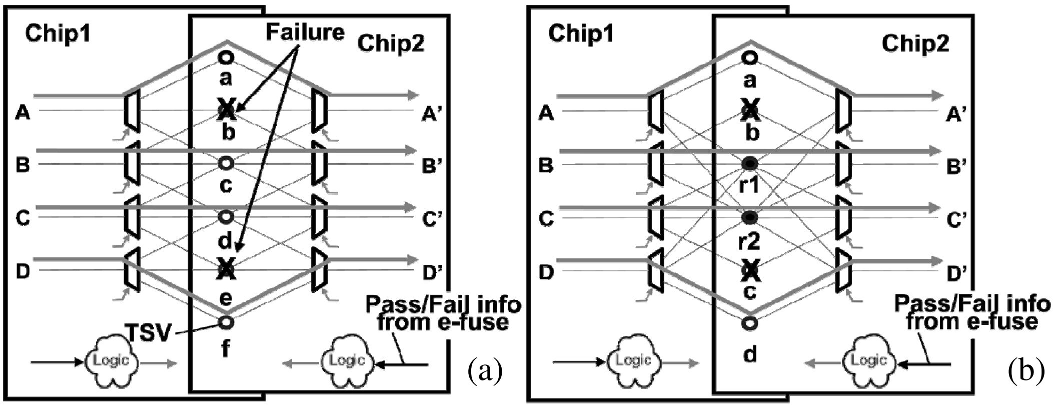


Figure II-9 TSV check and repair strategy for 4 signals A,B, C, D [KCH10]

TSVs are tested externally for open and short faults and using scan chains which output diagnosis vectors to external diagnosis circuitry. Faulty TSVs are repaired by their neighbors by programming arrays of e-fuses. Thus, detour paths decrease, reducing interconnect routing complexity and capacitive loads. For TSV technologies with *80μm* pitch, the authors report a *~2%* chip area overhead due to the total footprint of spare TSVs.

In [ZKA11], the grouping strategy was used within TSV bundles in order to efficiently use spare resources for different TSV defect distributions. Their experimental results showed that having fewer groups and more spares pays-off for high failure rates. The authors of [JXE12] use the *row/column repair* strategy for regular arrays of TSVs. Hence, for each $N \times M$ TSV array an extra row / column of spares (i.e. $N+M$ spares) is added such that faulty spares are replaced by their nearest neighbor. Using specific repair algorithms, they can achieve better yield with fewer spares than existing group-based solutions.

Although manufacturing defects remain the main source of faults in 3D ICs, in-field failures and transient errors during normal system operation could compromise the entire system. In the following, existing reliability enhancement solutions specific to 3D ICs are presented.

2.2.3.3 *Reliability in 3D ICs*

Stacking silicon dies causes major thermal issues that have a dramatic impact on reliability. It is expected that higher temperature gradients inside 3D chips accelerate the TSV fault rates due to electro-migration [EK08]. Solutions to this issue include adding thermal vias [LHZ06] or micro-channel liquid-cooling [BMR08]. Similar to 2D technologies, it is also possible to avoid hot spot formation in silicon layers containing processing cores by careful thermal-aware task mapping. For 3D systems, this solution was explored in [ZGS08, CAA09]. In [CAR10], an energy-efficient hardware-software solution based on liquid cooling and task mapping was proposed.

Another reliability issue for 3D ICs is signal integrity during TSV traversal. In [GWP09, LSL11, LSL11a], it has been shown that TSV coupling has a negative effect on signal integrity. In this case, transient faults due to crosstalk are likely to occur during system operation, especially because we may use high frequency signals on TSVs, raising serious reliability concerns. A solution to this issue is to shield TSVs by interleaving functional TSVs with grounded vias [GWP09]. The disadvantage of this solution is that the number of TSVs per chip doubles. The authors of [LSL11] have found that if the distance between TSVs is increased then the effects of coupling are reduced and circuits perform better (i.e. shorter critical paths). However, increasing the minimal TSV pitch by *2.5μm* from *10μm* results in a design area overhead of *31%*. For area-sensitive designs, a placement-refinement strategy (i.e. TSV coupling-aware placement) is also proposed. In [LSL11a] it has been shown that TSV spacing is not always enough and two alternative solutions are proposed (i.e. shielding sensitive TSVs and buffer insertion), each with its advantages and disadvantages. In the first case, eight grounded TSVs (i.e. north, worth-west, west, south-west, south, south-east, east and north-east) are inserted around sensitive TSVs in order to minimize its interference with neighboring TSVs. In the second case, TSV drivers are over-designed in order to compensate crosstalk between neighboring wires.

Different chip-level solutions to the test, reliability and yield challenges of 3D ICs have been presented in the last sections. However, the main objective of this work is to address these challenges in the context of 3D Networks-on-Chip (3D NoCs). In the following section, an overview of 3D NoC design is presented along with test, yield and reliability improvement solutions that are specific to 3D NoCs or have been adapted from 2D NoCs.

2.3 *Networks-on-Chip in 3D Systems*

The challenge of communication-centric MPSoC design is to implement a scalable, energy-efficient global interconnect fabric. Connecting hundreds or even thousands of cores in an efficient way becomes possible using packet-switched networks (i.e. Networks-on-Chip) [BdM02, AT03]. In NoC-centric designs, *IP* blocks are interconnected and communication is performed using specific system-level protocols such as VCI/OCP or AMBA AXI.

For 3D MPSoCs, designing a scalable interconnect fabric poses several challenges. Adding the third (vertical) dimension to the 3D NoC brings more flexibility, but challenges due to inter-die synchronization, low TSV reliability and yield must be addressed. In this section it is shown how the transition from 2D to 3D NoC can be made. It is worth noting that the design, test and reliability challenges of (2D) NoCs have been the subject of numerous papers in the last decade. Hence, some of these solutions have been (or could be) adapted to a stacked 3D integration setting.

2.3.1 From 2D NoCs to 3D NoCs

In transaction-based systems, communication is initiated by a *master* (initiator) node that sends a request to a *slave* (target). After the request is processed, a response is returned to the *initiator*. This form of communication is more common in MPSoCs for mobile and high performance embedded applications than connection-based communication, which consists in explicit message passing between IPs. In Figure II-10, a transaction between an *initiator* and a *target* is represented.

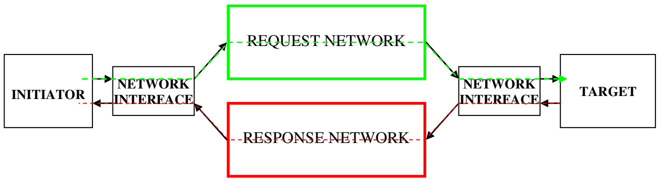


Figure II-10 IP blocks with initiator and target interfaces connected by a NoC

A common technique in transaction-based communication is to have two physically separated networks for requests and responses. This solution has the advantage that it guarantees high-level deadlock freedom (i.e. deadlocks at transaction-level) [HGR07]. The request message of the *initiator* IP, which is packetized by the *network interface*, is routed through the *request network* to the *target* IP. The request message is recreated and it is processed by the local *target* IP. The response message of the *target* IP is packetized by the network interface and transmitted to the initiator through the *response network*. At this point, the initiator can decide whether the protocol ended correctly or not.

In MPSoCs, abstraction layers hide implementation details of different functionalities. In the example above, although the packetization, packet routing, and flow control mechanisms are hidden from the *IP* blocks, communication services are transparently available to them.

It is worth noting that there are no fundamental differences at system-level between 2D and 3D NoCs: *IP* blocks exchanged messages using store/load transactions. However, the IP blocks of 3D systems are distributed across the stacked dies. Therefore, there are many design solutions for the global communication fabric. Let us consider two dies for a stacked 3D system. In Figure II-11, two possible 3D interconnect fabrics are represented.

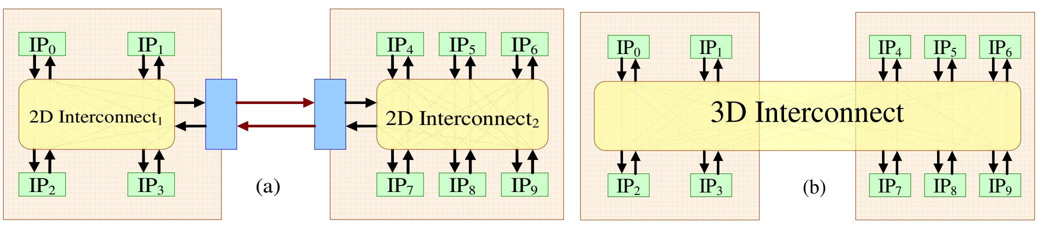


Figure II-11 Interconnect fabric strategies for stacked 3D SoCs

In the first case, a 2D fabric interconnects IP blocks within each layer and *die-to-die* communication is performed via dedicated interface blocks (e.g. Low-Latency Interface from MiPi Alliance [LLI]). The advantage of this approach is that each layer can be designed independently, with different communication protocols, network architectures (e.g. topology, data size, link-level synchronization) and even CMOS technology (i.e. *heterogeneous interconnect fabric*). Moreover, link-level synchronization and serialization issues of inter-die communication are hidden from the intra-die communication fabric. However, the *off-die* communication module has the difficult task of ensuring functional compatibility between the intra-die NoCs.

Alternatively, it is possible to implement an interconnect fabric that spans across multiple layers (see Figure II-11 (b)). In this case, all IP blocks implement the same communication protocol and the performance penalties due to protocol conversions are eliminated. However, inter-die synchronization and serialization must be addressed at lower abstraction layers (data link and physical). In the remaining of this thesis, only such NoC interconnect fabrics are considered for 3D MPSoCs. In the following section, some design considerations of 3D NoCs are presented.

2.3.2 3D NoC Design

3D NoCs have their routing nodes distributed across the stack and they have a mix of intra-die/inter-die links. The 3D NoC ensures inter-die and intra-die communication of the IP blocks. Depending on the IP block distribution, the 3D NoC may have different topologies. For example, Figure II-12 shows an MPSoC with *N* IP blocks that are connected using 3D NoCs with regular (a), quasi-regular (b) or custom topologies (c).

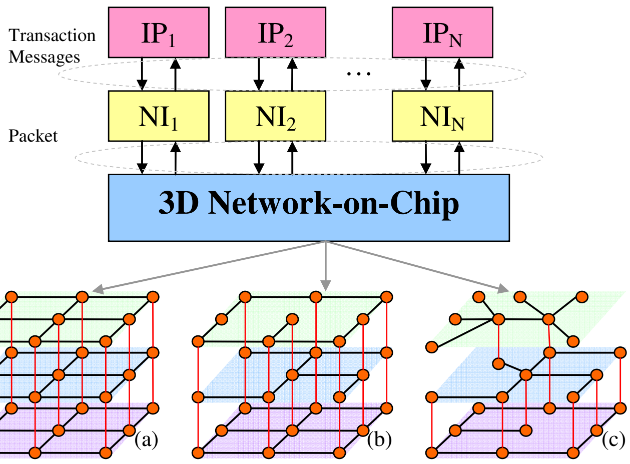


Figure II-12 MPSoC implemented using 3D NoCs

Routers are responsible for forwarding incoming packets in the requested direction. The header flit of each packet is checked first in order to determine the next hop. The header is immediately forwarded in the requested direction and subsequent flits are sent as they arrive, in a pipelined fashion. Extrapolated to all routers along a path, one packet can span across a number of routers (i.e. it *worms* its way through the network). From the circuit implementation perspective, 3D NoC routers are planar structures (i.e. 2D circuits) with interfaces to inter-die links. Although decomposed 3D router architectures that span across multiple layers have been proposed in [CNP07, PED08], such implementations require very high density TSVs that are not available in the near future technologies [MZ09]. Hence, in this thesis it is considered that 3D routers are planar circuits with ports for intra-die and inter-die communication.

Because 3D NoC topologies expand in the vertical dimension, their connectivity is higher and the network diameter (i.e. maximal distance between any two routers) can be reduced. It has been shown that, compared to 2D NoCs, the performance metrics (i.e. latency and throughput) are significantly improved [PF07, FP09, QLD09]. In [FP09], the performance improvements of 3D NoCs were assessed for 3D mesh, 3D thorus, ciliated mesh, and hybrid network-vertical bus architectures. TSVs are an expensive resource, as their footprint is non-negligible. NoC-based MPSoCs with custom / application-specific 3D topologies that optimize TSV utilization, power consumption and system performance have been proposed in [YL08, SMB09].

The TSV count can be reduced by implementing serialization techniques for inter-die links [Pas09, DVS11]. In [Pas09] it was shown that this approach leads to minor penalties on system performance: for the SPLASH benchmark, the performance penalty is less than 5% when all vertical links perform *64:1* serialization. An alternative solution was proposed in [BSS08], where a series of 3D NoC topologies with partial vertical connectivity have been proposed. Their results have shown that reducing the number of inter-die links has a minor impact on system performance.

Synchronization is another important aspect of 3D NoCs, as correct clock distribution across the silicon stack is very difficult. This issue was addressed in [LAB08], where an existing link-level mesochronous synchronization mechanism was implemented in the inter-die links of 3D NoCs.

In order to cope with synchronization, power and variability issues, the author of [TVC10] propose a fully asynchronous 3D NoC architecture. This network architecture uses asynchronous serialization to reduce the number of TSVs / chip. Moreover, in order to reduce router complexity, a hierarchical architecture was proposed. Hence, complex 7-port routers are decomposed into more simple 5-port and 3-port sub-routers.

Testing and fault-tolerance of NoCs are topics that have been addressed by the research community. Although most of this work was developed for 2D NoCs, it can be implemented in a 3D integration framework. In the following, the main contributions to NoC testing and fault tolerance are presented.

2.3.3 NoC Testing

Most test efforts in the SoC paradigm are focused on testing IP blocks. The shift to communication-centric design styles (i.e. NoCs) raised the question on how to test the system interconnect fabric. NoC test strategies must address two main problems: testing individual routers and testing links. Although this thesis addresses link testing (i.e. inter-die links are the particularity of 3D NoCs), router test strategies are presented for completeness.

The authors of [VDG03] suggested that the NoC can be considered a simple IP block with two particularities: it is composed of many identical sub-cores (i.e. links, routers, and network interfaces), and it occupies a central position in the system, its role being to interconnect other IP blocks. In NoC-based systems, the NoC is tested first and, if faulty components are detected, then the chip is discarded or sent for further diagnosis. The NoC regularity allows the re-use of test data for all identical components (e.g. routers, links).

The link and router test issues were also addressed in [GIS07]. Tested links/routers are reused for transporting test data to components-under-test in a recursive manner. The authors have also proposed a test parallelization mechanism based on multi-cast protocols such that test data is simultaneously sent to multiple routers/links-under-test. The external tester accesses the NoC through a network interface and the off-chip generated test vectors (i.e. test packets) are sent through the network.

Because NoCs are relatively complex structures that spread across the chip, different Built-In-Self-Test (*BIST*) strategies have also been used for testing NoC routers and links [GPI06, GIS07]. Hence, routers and links are tested periodically or under external control. In [GPI06], the Maximum Aggressor Fault (MAF) model proposed in [CDB99] is used for implementing interconnect BIST strategies for NoC links. In [GIS07], the authors have shown that traditional BIST implementations for FIFOs are not efficient for NoCs, and they propose a distributed BIST architecture.

In [ZGB10], a complex BIST architecture is used for testing the DSPIN NoC [PG07] communication channels (i.e. router FIFOs and inter-router short wires) and internal router components (i.e. routing logic, FSM, and intra-router long wires). This off-line BIST strategy is then used for initializing the 2D mesh DSPIN NoC by deactivating faulty components.

In [GIS06], on-line fault detection and location based on code-disjoint elements and parity check were proposed for NoC routers and links. During NoC normal operation, faults are detected by encoding flits at router inputs and comparing the results at its outputs. If faults are detected then fault containment and recovery actions are taken.

It is also important to have test capabilities during system life-time. Therefore, a series of software-based test and diagnosis strategies have been proposed. The authors of [KNG09] proposed a mechanism for discovering fault-free paths between a specific I/O port and fault-free processor cores. The result of this process is a fault-free communication map, which is used for finding the NoC fault-free components and configuring them. This centralized solution is driven by the complex configuration master (i.e. smart I/O port). Thus, this method induces a weak point into the chip, as the I/O port is the critical resource.

The *Distributed Cooperative Configuration Infrastructure* (DCCI) [ZRG11] uses embedded Configuration Firmware (CF) in each cluster of the MPSoC. This fully-distributed solution has the benefit that the fault-free communication map is built locally by exchanging messages with neighboring nodes. Nodes that present faults are considered *black-holes* and their location is reported in order to define a configurable fault-tolerant routing algorithm that avoids them.

2.3.4 Reliability and Yield

In VDSM technologies, soft-errors (i.e. single-even transients SETs, single event upsets SEUs), power grid fluctuations, power supply noise, electro-magnetic-interference (EMI) are contributing factors to increasing error rates. Higher clock rates, lower voltage levels and increasing inter-wire coupling exacerbate crosstalk on wires. NoC transmission errors, due to transients on links and routers, affect system reliability and may lead to system failures.

Routers and links can also be affected by permanent faults due to manufacturing and aging / wear-out defects. Thus, fault-tolerance capabilities must be included in NoCs in order to ensure reliable communication. In this section, an overview of NoC fault tolerance strategies is presented.

2.3.4.1 *Fault-tolerant routing*

At network-level, router and link permanent faults are mitigated using fault-tolerant routing algorithms. When router and link failures are taken into account, even regular topologies manifest some degree of irregularity. In such cases, even simple algorithms become more complex, as they must find alternative source-destination paths around faulty components. Although routing based on (reconfigurable) routing tables is always a possible fault-tolerant solution even for irregular topologies, it is difficult to develop optimized approaches [AN05].

An important feature of fault-tolerant routing algorithms is that it must not deteriorate network performances in the case when there are no faults. Usually, a simple routing scheme is used in regular topologies in the fault-free case. After faults appear, fault-tolerant routing is used. Switching between deterministic and adaptive routing is also exploited to reduce congestions and reducing the overall routing overhead [HM04].

The simplicity and regularity of 2D NoC topologies (i.e. mesh, thorus) enabled the implementation of efficient fault-tolerant algorithms [ZGT08, FDC09, RFR10]. These fault-tolerant routing algorithms have been designed only for 2D mesh/torus topologies, based on different approaches such as faulty blocks, turn models, intermediary nodes, virtual channels or networks etc. [CA95, CC01, HS04].

Another type of NoC fault-tolerant communication based on redundant messages has also been proposed in [DM03]. Instead of sending a unique message from source to destination, multiple copies of the same message are sent on different paths. Packets traversing faulty components and all but the only first correctly received packet are dropped. Fault-tolerant communication algorithms are analyzed in [PLB04]: two different flooding algorithms (gossip and direct) and one random walk algorithm are investigated. The results of this analysis show that the flooding algorithms have an exceedingly high communication overhead. The redundant random walk overhead is significantly reduced, while useful levels of fault-tolerance are maintained. From the energy consumption point of view, the random walk strategy proved to be considerably more efficient.

2.3.4.2 *Link repair*

Link-level *spare-and-replace* strategies consist in replacing faulty wires with functional spares. The interconnect yield of 2D NoC links can be significantly improved with minor costs by adding spare wires and a balanced configuration fabric [GIS06a]. Unfortunately, it is not always possible to include enough spares to ensure high reparability. This problem was addressed in intra-die interconnect technologies by link-level serialization.

Time redundancy with error correction codes (i.e. Hamming single error correction codes) is proposed in [TLP07] for self-timed on-line reconfigurable intra-die links. When permanent or intermittent faults are detected and no functional spare wires are available, data is split, duplicated and sent encoded in two transmission cycles. The link uses single error correction (*SEC*) codes to correct transient errors and to detect on-line interconnect faults. In this syndrome store-based (*SSB*) detection scheme, the error indicator vector of the Hamming correction block is monitored in order to detect intermittent or permanent faults (i.e. error correction is performed for the same position).

The interconnect variability of NoC links is addressed in [HSS09] by a technique called *phit reduction* (PR). This technique can be implemented at link-level where data is sliced in smaller data blocks that are serially transmitted. At network-level, *PR* consists in reducing the data size for the entire network. Packets that are expected to traverse partial faulty links are further serialized (i.e. each flit comprises fewer bits than the normal flit size), causing the packet length to increase considerably. This serialization / deserialization process is performed at source / destination by reconfiguring the network interfaces. The experimental results of [HSS09] showed that the local (link-level) *phit* reduction strategy is significantly better in terms of performance and power than the global (network-level) solutions.

Partial faulty links with application-specific routing are jointly used in [PKC10] for high performance and energy efficiency in 2D NoCs. It was shown that when they are jointly used, they have very little impact on the system performance. The link-level fault tolerant mechanism of [VSN10] addresses the problem of on-line fault detection and diagnosis and uses *partial faulty links* between the routers. The scheme relies on the fault diagnosis vector, which indicates the position of the faulty wires, to create at the link interfaces *mask* vectors. These vectors indicate which data bits are received on the functional wires. Experimental results using the SPLASH benchmark have shown that performance penalties are less than *5%* even for high interconnect failure rates.

2.3.4.3 *Signal encoding*

Coding on-chip signals is a powerful technique that can solve the delay, power and reliability problems of on-chip wires. The power dissipation on wires can be reduced by static or adaptive low-power coding schemes [KBS00, ZLS02] that reduce the self-/coupling-transition activity. Most low-power codes are not linear and they are complex and lead to significant overheads, making them less effective for relatively short interconnects like on-chip busses or NoC links [KNM04]. Crosstalk avoidance codes [DTK01, VK01, SAS04, DZK08] reduce the worst-case delay by ensuring that a transition from successive code words does not cause transitions on opposite directions on adjacent wires. The simplest methods to satisfy this condition are to add grounded lines between active lines or duplicate data wires. The number of additional lines increases significantly for crosstalk avoidance codes because there are no linear codes that avoid critical switching patterns with less than 100% redundancy [SS05] (i.e. twice as many wires). Error detection (EDC) (e.g. parity, CRC, Berger codes) and correction codes (ECC) (e.g. Hamming, Hsiao) improve on-chip communication reliability. EDC and ECC are jointly used in complex coding schemes with improved error correction capabilities [SS05, GBB08].

Coding is also used with error recovery mechanisms (e.g. retransmission) in different error control scheme for flit and packet protection at different abstraction levels (i.e. data link, network and transport). In [BBM05], the AMBA AHB fabric of SoCs based on the LEON3 processor from the Gaisler IP Library is enhanced with error resilience. The AHB implementation is modified in order to accommodate the error detection and retransmission mechanisms. Every time errors are detected, the slave retry mechanism of AMBA AHB notifies the master (i.e. LEON3 cache controller) that it should retry the transaction (i.e. system-level recovery). If access to the bus is not lost, a new load/store transaction is initiated, resulting in potentially high performance loss. Error correction is also implemented, without affecting the AMBA AHB protocol and with minimal impact on transaction timing (i.e. 2 AMBA AHB clock cycles per transaction are lost for encoding and decoding). Although there are no system-level assessments, the authors have concluded that retransmission is more efficient than error correction. However, they have recognized the limitations of their study with respect to the performance evaluation and the communication fabric.

The authors of [MTV05] provide an in-depth analysis of different link-/network-level error control schemes for NoCs. Parity and CRC codes are used for network-level error detection on individual flits and packets, while error recovery is based on transport-level packet retransmission. In this scheme, individual flits are encoded using ECC and transport-level packet retransmissions are performed when multiple errors are detected. At link-level, flit and packet retransmission schemes are also considered. Their findings indicate that the packet-/flit-level scheme is more efficient both in terms of energy and performance. Similar results have been presented in [JLV05], where flit-level error correction encoding over the entire network is found to be more efficient than network-level packet retransmission. In [RAM07], data link-level and network-level error correction schemes using Hamming, Hsiao and symbolic codes have been proposed for Spidergon STNoC. Unlike previous approaches, the error control mechanisms are configurable, as the error correction modules can be bypassed if data integrity is not critical.

Error control affects system performance, as error recovery penalizes the overall network latency. Performability has been defined as a performance metric for dependable systems with performance degradation. In [EAR10], the performability/energy tradeoffs of NoC links using error detection, correction with retransmission are discussed with respect to noise power, time constraints, and wire length. The study has been carried of the three classes of link error control schemes: automatic retransmission query (ARQ), forward error correction (FEC) using single error correction (SEC) codes and hybrid error correction and retransmission (HYB) using Hamming SEC codes. The findings of that study can be summarized as follows. At the maximum voltage swing, the maximum achievable performability of error correction and retransmission (HYB) is always higher than what is achievable from detection and retransmission (ARQ), and correction (FEC). For a given performability constraint, HYB consumes less energy than ARQ and FEC, except for when short wires are used, or when tight time constraints are imposed. When short wires are used, HYB provides the best performability, but consumes the most energy, while FEC provides the least performability and consumes the least energy. Finally, when tight time constraints are imposed, HYB and FEC provide almost the same performabilities and can provide better performabilities than ARQ. However, FEC consumes less energy than HYB, making it the better choice.

2.3.4.4 *Robust router architectures*

Circuit-level fault tolerant solutions for sequential and combinational logic blocks [Nic99, AN00] may improve router and NoC reliability. Some of these techniques have also been implemented for NoCs [FKC06, PNK06, TMS07] in order to mitigate transient and delay faults in link and routers. Of course, these solutions can also be implemented for 3D NoC routers. Hardware redundancy can also be used at network level to improve system robustness. In this case, node faults due to manufacturing defects are mitigated using spare nodes [SC11]. Hence, functional spares replace faulty nodes by circuit-level network reconfiguration.

In [CPB06], *N*-modular-redundancy (NMR) techniques were used for improving router-level reliability. NMR approaches are expensive, since each component must be implemented *N* times. This solution may not be efficient at network-level, as some components are impossible or prohibitively expensive to duplicate. Moreover, when functional spares run out, the loss of a single router or link means the loss of the entire network. VICIS [DFB12] is a highly resilient NoC that maintains high reliability by leveraging the inherent router-/network-level redundancy. It provides higher reliability than NMR-based solutions with a *42%* area overhead. In VICIS, each router uses a BIST to diagnose hard faults and runs a number of algorithms to best use error correction codes (ECC), port swapping, and a crossbar bypass bus to mitigate them. Distributed algorithms are used for solving network-wide problems, protecting the network against critical failures in individual routers.

2.3.4.5 Extensions to 3D NoCs

Some of the above mentioned techniques have been implemented for 3D NoCs. At network level, fault-tolerant routing algorithms have also been applied to regular or quasi-regular 3D NoC topologies [RAA10, PZ11]. However, the hardware costs are non-negligible, as traffic traveling up and down the stack is separated on independent virtual/physical channels, in order to prevent deadlocks.

In [LML08], faulty TSVs of 3D NoCs vertical links are repaired by replacing them with functional spares. MUX-based crossbars reroute data and control signals on functional TSVs. These signals are generated off-chip and they are stored in one-time-programmable (OTP) memories. For 9.75 defects per million, the TSV yield is improved from ~67% to more than 99.99% by allocating two wires for each signal. The area overheads of the entire chip are reported to go up to *~2.1%* in 130 nm technologies and *~3.8%* in 65 nm. The results also show that, as the TSV pitch scalability is not compatible with that of CMOS, spare-based repair using grouping may lead to higher area overheads in advanced technologies.

2.4 *Conclusion*

This chapter overviewed the main challenges of 3D integration and focused on the on-chip communication challenge of 3D MPSoCs. The test, yield and reliability issues of 3D MPSoCs can be alleviated using innovative techniques that explore the particularity of 3D integration, or reusing well-established techniques.

Interconnect Built-In Self-Test (*IBIST*) strategies have the main advantage that no external intervention is required. In 3D MPSoCs, testing thousands and tens of thousands of TSVs becomes more efficient using *IBIST* implementations. Although existing solutions can be used for TSVs, they can be too conservative and lead to high overhead and long test times. In the following chapter, a *TSV-IBIST* implementation using the novel K^{th}*-order Aggressor Fault* (KAF) model is presented in Chapter 3.

Chapter Three

TESTING THROUGH-SILICON-VIAS IN 3D NOC LINKS

The inter-die links of 3D NoCs are crucial components, as most energy-efficiency and performance benefits are due to them. In this chapter, the inter-die link test issues are addressed by an Interconnect Built-In Self-Test (IBIST) strategy. The proposed test strategy sensitizes structural permanent faults like opens and shorts, and also delay faults due to crosstalk. One of the fault models that can sensitize such faults is the well-known Maximum Aggressor Fault (MAF) model. However, this model is too conservative and it leads to long test sequences and non-negligible hardware costs. Therefore, an alternative solution is proposed: the K^{th}-Aggressor Fault (KAF) model. In the new model, the aggressors of victim wires are the neighbor wires within a distance order K. The aggressor order K is technology-dependent and is determined such that the test times are minimal and the fault coverage is maximal. A configurable KAF-based IBIST implementation where tests can be performed using different aggressor orders K is also presented. Although the configurable IBIST area is significant, interconnect tests during system lifetime can be performed using lower aggressor orders, significantly reducing test duration.

3.1 *3D NoC Inter-die Interconnect BIST*

Interconnect Built-In Self-Test strategies may be used for testing the TSVs of inter-die links in 3D NoCs. The major advantages of an *off-line Interconnect Built-In Self-Test* (*IBIST*) strategy are the capability for at-speed testing and minimal external intervention. In this section, the *IBIST* architecture for TSVs is described along with existing test pattern generation strategies.

3.1.1 ***Testing TSV faults and defects***

Void formation, lamination due to thermally induced stress, height variation, and XY misalignment are the major TSV failure mechanisms [LCF07, KXM09, LCD09]. Stacking silicon dies causes major thermal issues that have a dramatic impact on reliability. It is expected that higher temperature gradients inside 3D chips accelerate the TSV fault rates due to electro-migration [EK08]. Moreover, in [GWP09, LSL11, LSL11a] it has been shown that TSV coupling has a negative effect on signal integrity.

The TSV failure modes are not fundamentally different from on-chip and board-level interconnects. In all cases, interconnect manufacturing and aging/wear-out defects are modeled using the basic faults: *open*, *short*, *stuck-at*, and *delay* faults. These faults are permanent, as they affect the interconnect behavior always in the same way. In order to illustrate the delay faults due to crosstalk, the aggressor-victim scenario is considered.

Signal transitions on a single wire are affected by the crosstalk noise induced by transitions on *all* other wires. In Figure III-1, these faults are summarized for victim wire W_2 and aggressors W_1 and W_3.

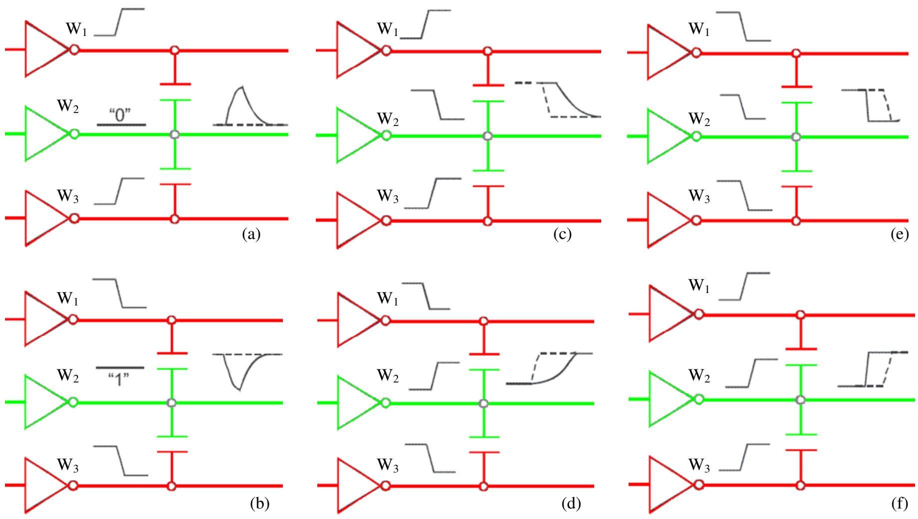


Figure III-1 Fault due to crosstalk in the aggressor (W_1-W_3) – victim (W_2) scenario

If the victim carries the constant signal '0' / '1' then the rising / falling transition on the aggressor TSVs causes a small voltage glitch on the victim wire, see Figure III-1 (a,b). If this glitch is large enough then it can be captured by a flip-flop. If transitions on victim and aggressor wires are in opposite directions then the rising / falling victim transition is delayed by the falling / rising aggressor transition, as shown in Figure III-1 (c,d). When the propagation delay is integral in designing setup and hold times, speed up of the signal from nominal delay is possible, giving rise to two more possible error conditions. The two faults, called rising and falling speed-up, are represented in Figure III-1 (e,f).

Although solutions like TSV shielding, spreading and buffer insertion have been proposed in order to reduce the effects of TSV coupling, they are not complete. Defects and parametric variations may affect the TSV coupling and the susceptibility to crosstalk noise. Worst-case design solutions that compensate delay faults due to crosstalk often prove to be too conservative and lead to significant costs. It is known that, for on-chip interconnect, it is more efficient to use less conservative design rules for crosstalk between wires and perform interconnect tests for delay faults due to crosstalk [CDB99]. Similarly to on-chip interconnects, TSV tests can be performed in order to detect crosstalk delay faults due to manufacturing and in-field failures during system life-time.

3.1.2 Interconnect BIST architecture

One of the advantages of *IBIST* is that tests are performed at nominal clock rates (i.e. at-speed tests). Hence, open, shorts and also delay faults (and delay faults due to crosstalk) can be detected. In NoC designs, it is very common to have unidirectional router-to-router links. Although NoC bidirectional links can be implemented, the extra costs required at the router interfaces makes them less attractive. Let us consider 3D NoCs with unidirectional links of width *N*. In Figure III-2, the simplified *IBIST* test architecture is represented for the inter-die link that connects the lower-die transmitter router T_X and the upper-die receiver router R_X.

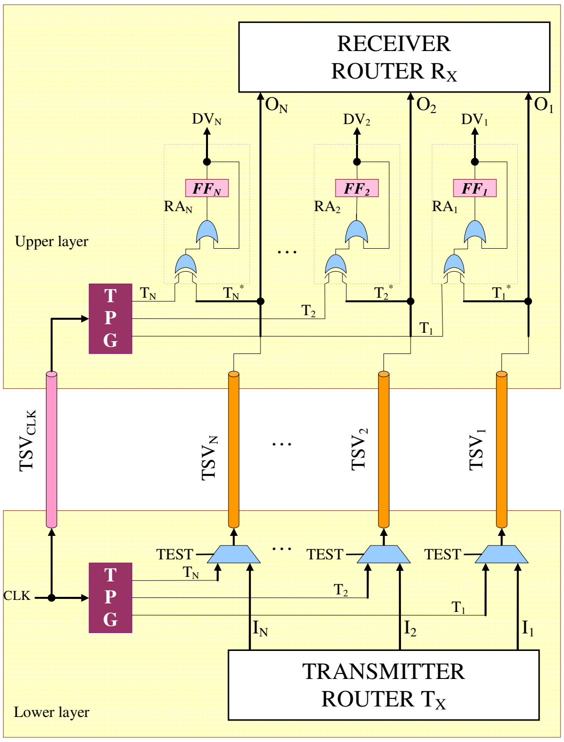


Figure III-2 Inter-die Link Interconnect Built-In Self-Test (IBIST)

During the test phase (i.e. *TEST='1'*), the T_X router interface is disabled (i.e. no new data can be sent) and *N* bits test vectors T_1:T_N are generated in the lower die by Test Pattern Generators (*TPGs*) and transmitted on TSVs. In the upper layer, the received vector T_1^*:T_N^* is compared to the original transmitted vector generated by a second *TPG*, identical to the one in the lower layer. During tests, the response analysis cells (*RA*) compare the response and test vectors and iteratively build the diagnosis vector DV_1:DV_N. In each response analysis cell RA_i, a *XOR* gate compares the received test stimulus T_i^* with the expected one T_i. Using an *OR* gate, any mismatch of these signals is stored in *RA's* internal flip-flop F_i. At the end of the test phase, the diagnosis vector which identifies the faulty TSVs is stored in *RA*'s internal flip-flop FF_1:FF_N. In order to improve circuit timing, intermediate retiming stages may be added. In this case, the upper die TPG must be delayed by a number of clock cycles equal to the number of the retiming stages. During the operational mode (i.e. *TEST='0'*), the timing of the link is slightly affected by the 2:1 MUXes delays added on the I_N:I_1 to O_N:O_1 signal propagation paths. However, for most implementations, this impact is negligible.

One of the challenges of implementing *IBIST* techniques is to ensure good inter-die synchronization. This is not an issue for 3D MPSoCs with Intellectual Property (IP) blocks implemented in 2D technologies with different voltage/frequency domains. However, the global interconnect fabric, which spans across the silicon layers and uses TSVs for inter-die communication, must be synchronous or mesochronous (i.e. the same frequency, but different phase). For fully-synchronous NoC fabrics, no synchronization is needed for the *IBIST* scheme. For mesochronous vertical links, communication is implemented using link-level clock synchronizers [LAB08] or multiple-clock FIFOs. In both cases, two clocks are used in the destination layer (i.e. upper layer in Figure III-2) for signal synchronization. Hence, the *IBIST* remains in the fully-synchronous domain of the transmitter die (i.e. lower die in Figure III-2).

To sensitize all types of faults due to crosstalk on NoC links, the Maximum Aggressor Fault (MAF) model [CDB99] has been used for NoC link *IBIST* [GIS07]. However, the implementation costs and complexity are non-negligible (e.g. more than 30% area overhead for a 2D five-port router with virtual channels [GIS07]), as *8N* test patterns are necessary to sensitize the faults for N wires.

In the case of TSV tests, inter-die link *IBIST* complexity can be reduced by restricting the number of aggressor TSVs to the nearest neighbors of the considered victim wire. It is important to note that the crosstalk-induced delay faults on TSVs are mainly due to the coupling between neighboring TSVs and not all TSVs connecting two dies. The proposed solution, which is presented in the following section, is the K^{th}-order Aggressor Fault (KAF) model that takes into account the distance between wires. Depending on the TSV spatial distribution, potential aggressors are classified in classes (i.e. *K*-orders).

3.2 *K^{th}-aggressor Fault Model for TSVs*

The K^{th}-Aggressor Fault (KAF) model enables crosstalk-induced delay fault detection by partitioning TSVs in aggressors and victims using the aggressor-victim scenario. In this section, the TSV partitioning in aggressor and victim set is presented for the K^{th} aggressor orders.

3.2.1 Defining aggressor orders

In 3D chips, vertical wires are distributed such that they do not affect the functionality of active logic blocks, facilitate heat removal and limit intra-die wire routing congestion. TSVs connecting two stacked layers may be uniformly distributed on regular arrays (e.g. *MxM*) or they may have non-uniform (irregular) distributions. In the proposed fault model it is considered that, for a given TSV distribution, the aggressors of a victim TSV are all the TSVs within a distance given by the minimal TSV pitch *p* and an arbitrary constant *K* (i.e. aggressor order). In Figure III-3, the first (*K=1*), second (*K=2*) and third (*K=3*) order aggressor of a victim TSV are represented for regular and non-uniform distributions.

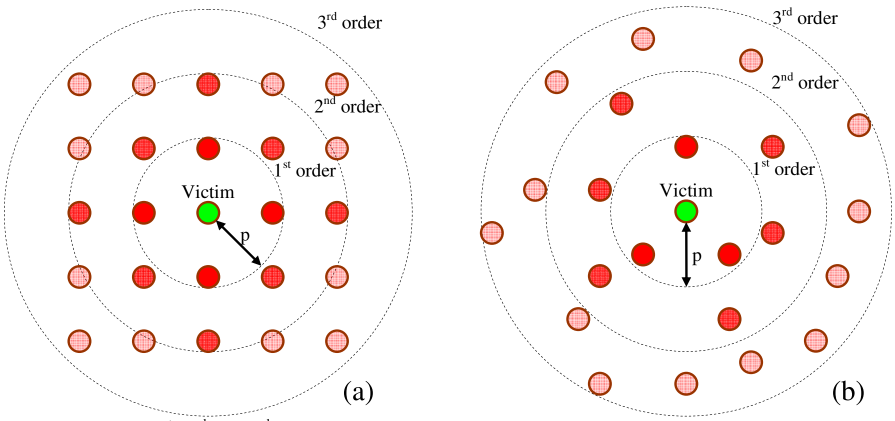


Figure III-3 1st, 2nd and 3rd aggressor sets for regular (a) and non-uniform (b) TSV distributions

The first order aggressors are all the TSVs within a range *p*; the second order aggressors are all the TSVs that can be found between *p* and *2p* distance away and the third aggressor set consists of TSVs that are more than *2p* away from the victim TSV, but less than *3p*. In general, the K-order aggressors of a victim wire and all TSVs within a distance range *p·(K-1)* and *p·K*. Note that the minimal pitch *p* is a technology-dependent constant that could partially compensate crosstalk effects between neighboring TSVs.

For a given aggressor order *K*, the victim TSV set is defined as a collection of TSVs such that transitions on any TSVs within the same set do not affect each other. In other words, each set contains TSVs such that no TSV in the set can be considered a *K*-order aggressor for other TSVs in the set (i.e. the distance between every pair of TSVs is more than $K \cdot p$). In the following, an algorithm for determining the TSV victim set of a TSV bundle is presented.

3.2.2 TSV partitioning in victim sets

The challenge of KAF-based pattern generation process is to correctly identify the victim sets. In the case of TSVs, this process is straightforward, as only the distance between TSVs has to be considered. In other words, the only criteria needed for including two wires TSV_i and TSV_j in different victim sets is the distance between them $Dist(TSV_i,TSV_j)$. For *N* TSVs and an aggressor order *K*, the victim sets $V_1 \ldots V_w$ are determined using the algorithm in Figure III-4.

```
1   S = 0 ;
2   for i=1 to N do Set(TSVi)=0;
3   while(notCovered())do
4       S++;
5       VS=ϕ;
6       for i=1 to N do
7           Free(TSVi)=true;
8       for i = 1 to N
9           if(Free(TSVi)and(Set(TSVi)==0)) then
10              Set(TSVi)=S;
11              VS= VSU{TSVi};
12              for j = 1 to N do
13                  if((i!= j)and(Dist(TSVj,TSVi)≤p·K) then
14                      Free(TSVj) = false;
15                  end if;
16              end for;
17          end if;
18      end for;
19  end while;
```

Figure III-4 TSV partitioning algorithm using the K^{th} order aggressors for N TSVs

Initially, no victim subsets exist and the TSVs are not attached to any group, as shown in lines 1-2. The *notCovered()* function returns *true* if at least one TSV not attached to a victim subset exists (i.e. there is at least one *j* such that $Set(TSV_j)=0$). In line 5, the current victim subset V_S is initialized for a given victim subset index *S*. In line 9 the first wire TSV_i, which is not attached to any victim subset (i.e. $Set(TSV_i)=0$) and not in the aggressor set of any wire in the current victim subset (i.e. $Free(TSV_i)=true$), is determined. In lines 10-11, if such a wire exists then it will be included to the current victim set V_S (i.e. $Set(TSV_i)= S$). In lines 12-16, all the aggressors of TSV_i (i.e. if TSV_j is aggressor of TSV_i then $Free(TSV_j)$ is *false*) are marked, in order to prevent including them in V_S. The aggressors are determined and marked in lines 13-15 using the distance *Dist()* function, the minimum pitch *p* and the aggressor order *K*. If there are unattached TSVs in the bundle then the victim subset index *S* increases in line 4 and the process above is repeated. The algorithm ends when each TSV belongs to a single victim set. In the worst case, this process takes *N* iterations, as the number of victim sets is smaller than the number of wires.

For regular and non-uniform TSV distributions in an inter-die link, the algorithm above determines the victim subsets. For a minimal pitch *p*, the TSV victim sets $\{V_1,\ldots,V_w\}$ are represented in Figure III-5 for the 1st aggressor order $K=1$. This example shows that, for a regular array, the 25 TSVs are partitioned in the $w=2$

victim sets represented in Figure III-5 (a). For the non-uniform distribution, the 23 TSVs are partitioned in *w=3* victim sets shown in Figure III-5 (b).

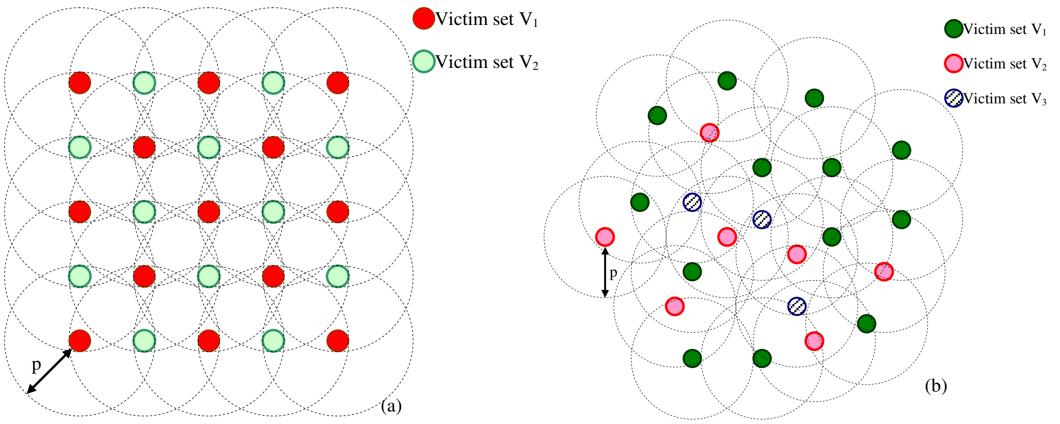


Figure III-5 The victim sets for 1st order aggressors in regular (a) and non-uniform (b) TSV distributions

Partitioning TSVs in victim sets can also be performed using graph coloring. In this case, the graph G has *N* vertices that correspond to the *N* TSVs. Given an aggressor order *K*, an edge *E* connects two vertices *i* and *j*, if the distance between TSV_i and TSV_j is less than $K{\cdot}p$. The vertex color represents the victim set of the corresponding TSV and the graph chromatic number (i.e. number of colors necessary to cover the graph) represents the number of victim sets *w*. Although both techniques give the same results, the graph coloring approach is more complex than the proposed partitioning algorithm in Figure III-4, as it often implemented using a backtracking strategy.

The KAF-based patterns must sensitize delay faults due to crosstalk between neighboring TSVs. However, as shown in Figure III-2, test patterns also traverse intra-die wires connecting the MUXes to TSVs, buffers and intra-die wires in the upper layer that connect TSVs to *RA* cells. Hence, faults on these components are also sensitized. Because all the ‘0’-‘1’ and ‘1’-‘0’ transitions are activated on every wire, all potential delay faults are sensitized. Open faults are also sensitized, since different signal values are transmitted on each TSV. Short faults are sensitized, but only those between aggressors and victims, since signals transmitted on TSVs of a victim set are identical. If there is a short between TSVs of the same victim set then this fault is not detected. However, victims of the same set are distant and the probability of such faults is very low, as shorts usually affect neighboring wires. Depending on the circuitry layout, a solution to avoid missing such faults is to use higher aggressor orders.

The main advantage of the KAF model over other similar models consists in reduction of the test times. For example, in the MAF model [CDG99] all neighboring wires are potential aggressors. In this case, each victim set contains a single TSVs and the test sequence is repeated *N* times for a bundle of *N* TSVs.

Using the KAF model, it is possible to test all TSVs within the same victim set in parallel. In the following, the implementation of KAF-based TPGs is presented.

3.3 *Implementation of KAF-based IBIST*

During the test phase, test patterns are generated both for aggressor and victim TSVs. The number of test sequences required to sensitize the six types of crosstalk delay faults is represented in Figure III-6 for an aggressor and victim TSV configuration.

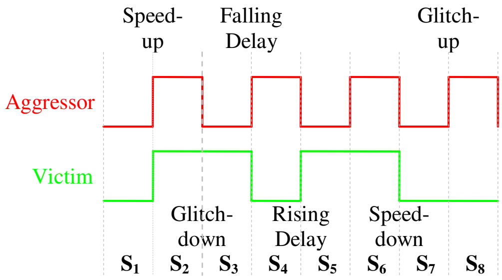


Figure III-6 Reduced test patterns for crosstalk-induced fault

Although there are six fault classes that are sensitized using *12* test vectors, the test sequence can be reduced to *8* vectors by rearranging and merging identical transitions. These test vectors can sensitize opens, short and delay faults due to crosstalk between one victim TSV and its aggressors. During tests, these signals are sent on each victim / aggressor TSV. In the remaining of this section two hardware implementations of KAF-based TPGs are detailed.

3.3.1 Generating KAF Test Patterns

Let us consider that the TSV bundle is split in *w* victim sets *{V1,V2,…,Vw}*. During TSV tests, each subset *Vi* identifies the victim wires, while the remaining TSVs are aggressors. During tests, the *w* victim TSV subgroups are tested by sending victim signals P_V on each TSV in V_i and aggressor signals P_A on the remaining TSVs. In Figure III-7, the KAF-based traffic pattern generator is presented.

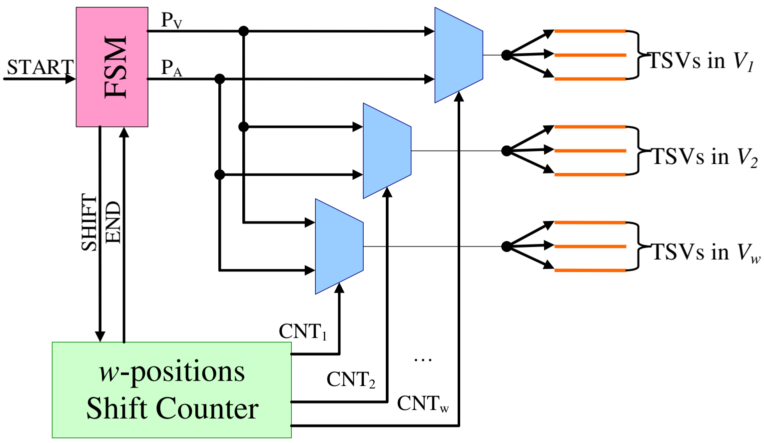


Figure III-7 Traffic Pattern Generator for KAF-based BIST

Shift counters with *w* positions CNT_1-CNT_w indicate the active victim subsets. For each victim set V_i, there is a 2:1 MUX that controls which of the v*ictim-aggressor* signals is mapped on its TSVs. If TSVs in set V_i are tested then CNT_i=*'1'* and P_V is transmitted on all TSVs in V_i. The other positions of the counter are *'0'* P_A is transmitted on the aggressor TSVs (i.e. TSVs not in V_i).

The *victim* and *aggressor* signals are generated using a *finite state machine* (FSM). The *FSM* has two inputs: *START* that initiates the *FSM*, and *END* that indicates the end of the victim set count. The *SHIFT* output of the *FSM* is an enable signal for the shift counter: when *SHIFT='1'* then the counter value shifts one position. The aggressor and victim signal values for the current state are generated (see Figure III-6) on the P_A and P_V outputs. In Figure III-8, the nine-state *FSM* that generate the shift control signal and the aggressor-victim signals is represented.

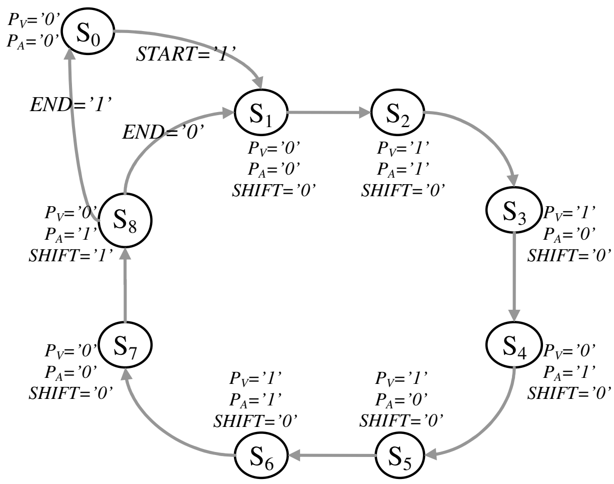


Figure III-8 FSM for generating aggressor / victim test signals

In the idle state *S0*, the inter-die 3D NoC link is in normal operation mode. When the interconnect test phase is initiated (i.e. *START='1'*), the shift counter indicates the first victim set *V1* and the *FSM* transitions to state *S1*. In this state, P_V*='0'* is sent on all victim TSVs in V_1 and *'0'* is sent on aggressor TSVs. Then, the *FSM* will perform transitions from *S1* to *S2,* then from *S2* to *S3* and so on, until it arrives in S_8. In each of these states, the enable signal of the counter is *'0'* and the aggressor-victim signal values are sent out on P_A and P_V, according to the eight vector values in Figure III-6. When the *S8* state is reached, the shift counter status is checked. If the shift counter has finished counting the victim sets, then the *END* signal is set to *'1'* and the *FSM* performs the *S8* to *S0* transition. Otherwise, for the currently active victim set V_i, the counter value increments, *SHIFT='1'* (i.e. CNT_i*='0'* and CNT_{i+1}*='1'*), and the *FSM* goes through the *S1-S8* states again for the next victim set V_{i+1}.

The delay of the KAF-based *IBIST* circuitry represented in Figure III-2 comprises the TPG delay (i.e. $\delta_{FSM}+\delta_{MUX}$), the 2:1 MUX delay δ_{MUX}, the interconnect and buffer delays δ_{WIRE}, and the RA cell delay $\delta_{RA}= \delta_{FF}+\delta_{XOR}+\delta_{OR}$. Hence, tests can be performed at high clock rates. For example, in a *65 nm* low-power technology, the *IBIST* can be clocked at *2 GHz*.

The difficulty in applying the KAF model is that the aggressor order must be determined for each TSV technology such that there is no fault coverage loss and the test times are minimal. An improvement of this implementation consists in modifying the test generation strategy with capabilities to perform tests for different aggressor orders. In the following section, a configurable KAF-based TPG implementation is presented.

3.3.2 Configurable KAF-based Test Patterns

Under-estimating the *K*-order reduces test duration, but fault coverage is reduced, as shorts between victim TSVs of the same set are not detectable. If higher aggressor orders are used then there are more victim TSV sets. However, the longer test sequence generated for higher aggressor orders may not sensitize more delay faults due to crosstalk. In this section, the implementation of configurable KAF-based *TPG*s is presented. The test logic can be configured during system lifetime with different TSV victim sets by adjusting the aggressor order with a value K ranging from K_{MIN} to K_{MAX}. The configurable KAF-based *TPG* architecture is shown in Figure III-9.

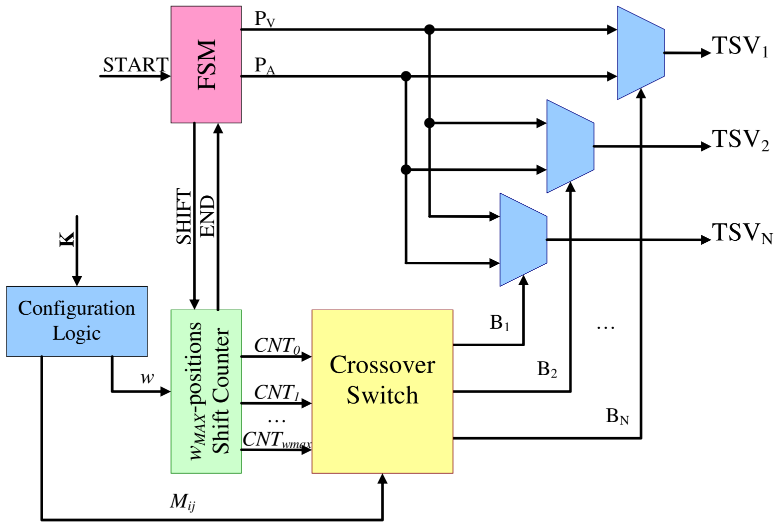


Figure III-9 Configurable KAF pattern generator

During interconnect tests, the aggressor or victim signal sent on each TSV is determined using shift counters that indicate the active victim set V_i. Given the maximal aggressor order K_{MAX}, the N TSVs are partitioned in maximum w_{MAX} of victim subsets (V_1,…,V_{vMAX}). Thus, a programmable shift counter with w_{MAX} positions (CNT_1-CNT_{wmax}) is used in order to indicate the active victim set. For an aggressor order K, the number of victim sets w is determined by the *Configuration Logic* block.

When the TSVs are partitioned in victim sets using the algorithm in Figure III-4, it is possible that some wires TSV_i are in the n^{th} partition for an order K_1 and in the m^{th} partition for a different order K_2. Thus, the behavior of TSV_i (i.e. value of signal B_i) is given by the counter's n^{th} position in the first case and on the m^{th} position in the second case. To illustrate this, let us consider a 4x4 TSV array. In Figure III-10, the victim sets are represented for the first $K=1$ (a) and second $K=2$ (b) aggressor orders.

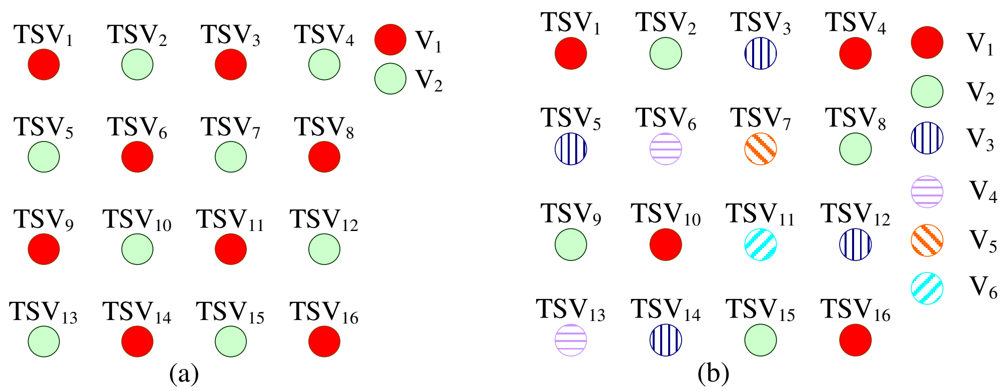


Figure III-10 Victim subsets for 4x4 TSVs for K=1 (a) and K=2 (b) aggressor orders

From these plots it can be seen that, for $K=1$ and $K=2$, some TSVs are in different victim sets. For example, TSV_5 is in the second victim subset for $K=1$ and it is in the third victim subset for $K=2$, while TSV_1 and TSV_{16} remain in the same group. The behavior of TSV_5 is given by the first position of the counter for $K=1$ and by the third position for $K=2$.

The counter position selection process is done for each wire TSV_i by a crossover switch whose inputs are the counter positions CNT_1-CNT_{wmax} and whose outputs are the TSV victim-aggressor control signals B_i. For each wire TSV_i, the victim signal P_V is sent if B_i='1', and the aggressor signal P_A is sent if B_i='0'. The crossover switch is implemented using MUXes or arrays of switching elements controlled by the matrix control signal M. For each valid CNT_i-B_j signal pair (i.e. there is an aggressor order for which TSV_j is in V_i) there is a unique select signal $M_{i,j}$. When $M_{i,j}$ is active, the input signal CNT_i is mapped on output B_j. The crossover switch control signals $M_{i,j}$ are determined by the *Configuration Logic* block. This block is implemented using truth tables that have the aggressor order K as input.

For the 4×4 TSV bundle, the crossover switch must be able to map the counter positions CNT_1-CNT_6 to the MUX selection signals B_1-B_{16} of each wire TSV_1-TSV_{16}. Let us consider that the *TPG* can generate test patterns for the 1st and 2nd aggressor orders. In this case, the signal sent on each TSV is determined by the counter register (i.e. the current victim according to the partitioning in Figure III-10). The mapping process of victim / aggressor signals on TSVs (i.e. MUX selection signals B_1-B_{16}) is performed by the crossover switch represented in Figure III-11.

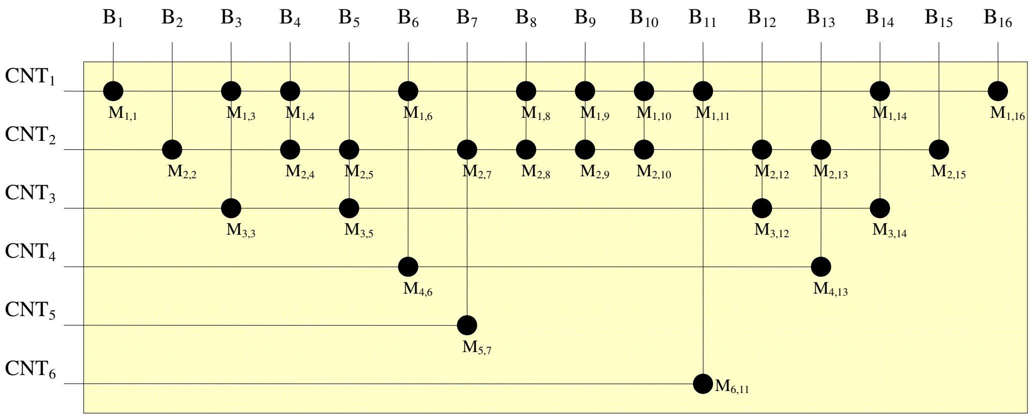


Figure III-11 Crossover switch structure for 4x4 TSVs having K=1 (w=2) and K=2 (w=6) aggressor orders

In the example above, TSV_5 is in the second and third victim sets for $K=1$ and $K=2$, respectively. In Figure III-11 there are two switching elements, one of which is active at a time, that connects B_5 to CNT_2 and to CNT_3, respectively. Similarly, B_{10} is connected to CNT_2 and CNT_1 such that $M_{2,10}$='1' for $K=1$ and $M_{1,10}$='1' for $K=2$.

A benefit of the configurable TPGs is reduced interconnect test duration during system lifetime. For example, initial tests could be performed with the highest aggressor order K_{MAX} in order to detect all possible errors. Then, in order to reduce link off-line duration due to tests, the aggressor order can be gradually reduced K_{MIN}. For example, M TSV interconnect test phases performed for an aggressor order K_1 require $8{\cdot}M{\cdot}w_1$ cycles, where w_1 is the number of victim sets. However, if m of these test phases are performed for a lower aggressor order K_2 then the test duration is reduced by $8{\cdot}m{\cdot}(w_1-w_2)$ cycles, where w_2 represent the number of victim sets for K_2.

The configurable KAF-based *IBIST* delay comprise the TPG delay (i.e. $\delta_{FSM}+\delta_{MUX}$), the 2:1 MUX delay δ_{MUX}, the interconnect and buffer delays δ_{WIRE}, and the RA cell delay $\delta_{RA}= \delta_{FF}+\delta_{XOR}+\delta_{OR}$. These delays do not comprise the *Configuration Logic* and crossover switch delays, as the input signal K is constant during the

TSV test phase. Therefore, high-speed configurable *IBIST* implementations are possible. For a 65 nm low-power technology, the configurable *IBIST* can be clocked at frequencies as high as *2 GHz*.

3.4 *Experimental Results*

In this section, the costs and test duration of KAF-based IBIST implementations presented in the previous section are estimated. The *IBIST* is implemented and evaluated for the 3D NoC unidirectional inter-die links that have TSVs arranged on regular grids. The costs are evaluated for a *65 nm* low power technology and a NoC working at a nominal *1 GHz* clock frequency (i.e. the nominal clock frequency of the 3D NoC seven-port router).

3.4.1 **Test duration**

Without loss of generality, let us consider an 8×8 regular TSV array with a $p_{TSV}=10\ \mu m$ pitch and the KAF constant $p=p_{TSV}$. In this case, the maximum aggressor order is $K=10$, as the maximum distance between any two TSVs in the 8×8 array is $\sim 99.4\mu m$, which represents less that $K=10$ TSV pitches $p=10\mu m$. The algorithm in Figure III-4 was used for determining the victim sets $V_1 \ldots V_w$ for any aggressor orders between $K=1$ and $K=10$. In Figure III-12 the number of test patterns (i.e. $8 \cdot w$, where w is the number of victim sets) is represented for different aggressor orders.

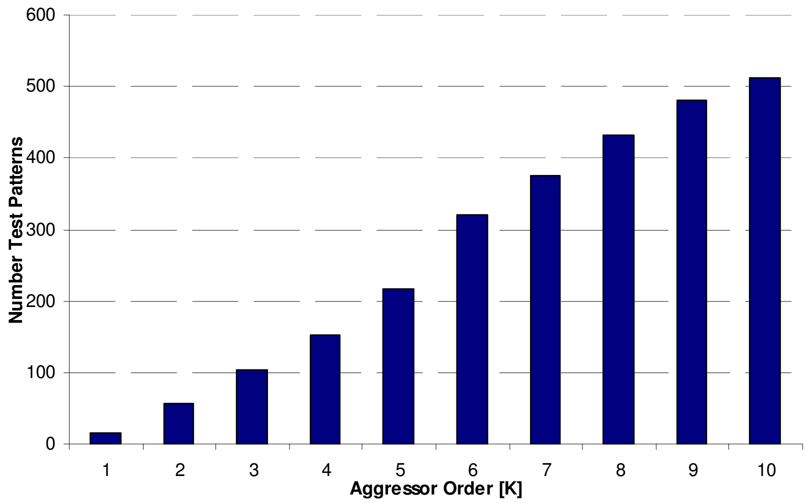


Figure III-12 Plot number of test patterns vs. K-aggressor order

When lateral TSV coupling is the main contributor to crosstalk-induced noise, tests can be performed for the 1st aggressor order (i.e. $K=1$). In this case, there are only two victim sets and *16* test patterns have to be generated. When the effects of high-order aggressors are taken into account, the number of victim TSV groups increases. The limit case, which corresponds to the MAF model, is when $K=10$. In this case, *64* victim sets (i.e. each TSV is alone in its victim set) are found and *512* test patterns are generated.

The *KAF*-based test patterns generated for the highest aggressor order are identical to the patterns generated using the *MAF* model, as the aggressors of each victim are *all* the other TSVs in the bundle. In Figure III-12, there is an almost linear dependency between the number of test patterns generated and the aggressor set. Hence, for a TSV technology, if the aggressor order is correctly estimated then the test duration can be reduced significantly. MAF-based tests have a linear dependency on the number of wires (i.e. for *N* wires there are *8N* test patterns). For KAF-based tests with a given aggressor order *K*, the number of victim subsets *w* determines the number of test patterns. To illustrate this dependency, $M \times M$ TSV arrays with different aggressor orders are considered. In Figure III-13, the number of test patterns for different aggressor orders *K* is given.

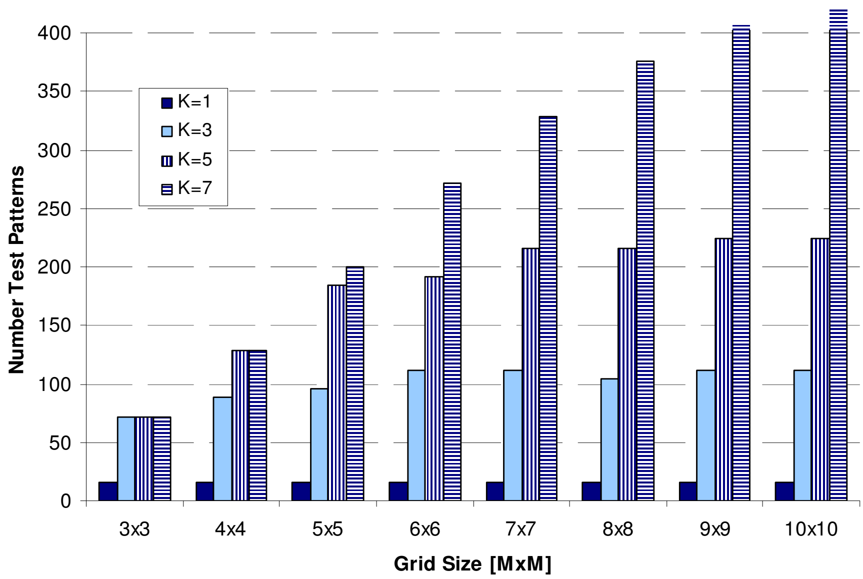


Figure III-13 Number of test patterns vs. K-aggressor order

These results show that, for small aggressor orders (i.e $K=1$ and $K=3$), the number of test vectors slightly changes with the $M \times M$ TSV grid size. For each victim TSV, the aggressors are bounded to a relative small area and there are few victim subsets. If the aggressors of each TSV represent only a small fraction of the total number of TSVs then the number of victim sets does not depend on the number of TSVs. When the aggressor order increases, a dependency of the test pattern count on the TSV count is observed. In this case, the aggressors of each victim TSV are bounded to a larger area and more victim sets are required to cover the TSV bundle. Hence, for smaller grid sizes (e.g. 3×3 and 4×4), the KAF-based *IBIST* is identical to the MAF-based IBIST (i.e. K ≥5), as the aggressors are all the remaining TSVs in the bundle.

3.4.2 KAF-based IBIST area evaluations

In the *IBIST* area assessments, the 3D NoC inter-die link test circuitry in the upper and lower dies (see Figure III-2) are synthesized in a *65 nm* low-power technology. In order to emphasize the benefits of KAF-based testing, MAF-based and marching-‘1’ implementations have also been considered.

3.4.2.1 IBIST

For the hardware cost evaluations, the *IBIST* modules on the transmitter side (i.e. lower die components: TPG and 2:1 MUXes) and the receiver sides (i.e. upper die components: TPG and RA cells) are considered separately. In Figure III-14, the area of the upper-die and lower-die *IBIST* components is estimated for an 8×8 TSV bundle with different aggressor orders.

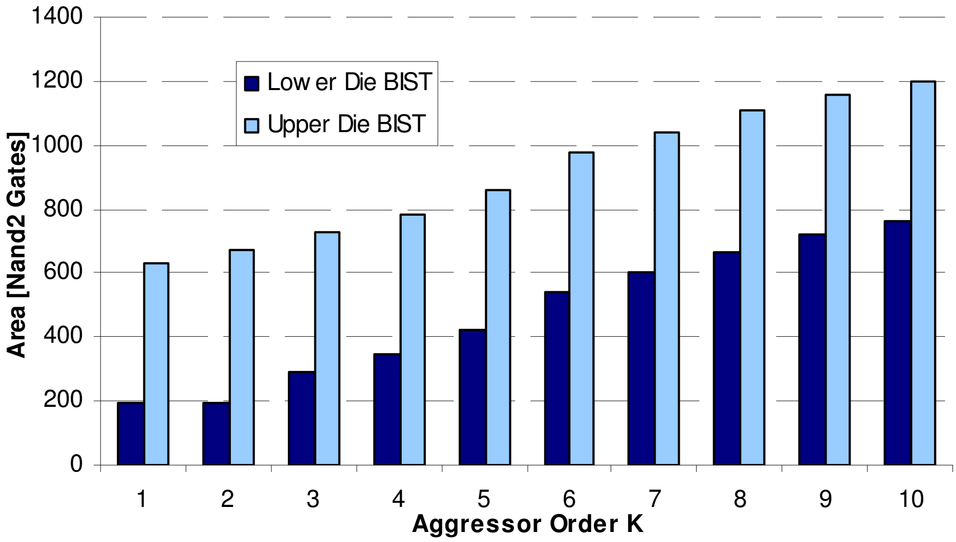
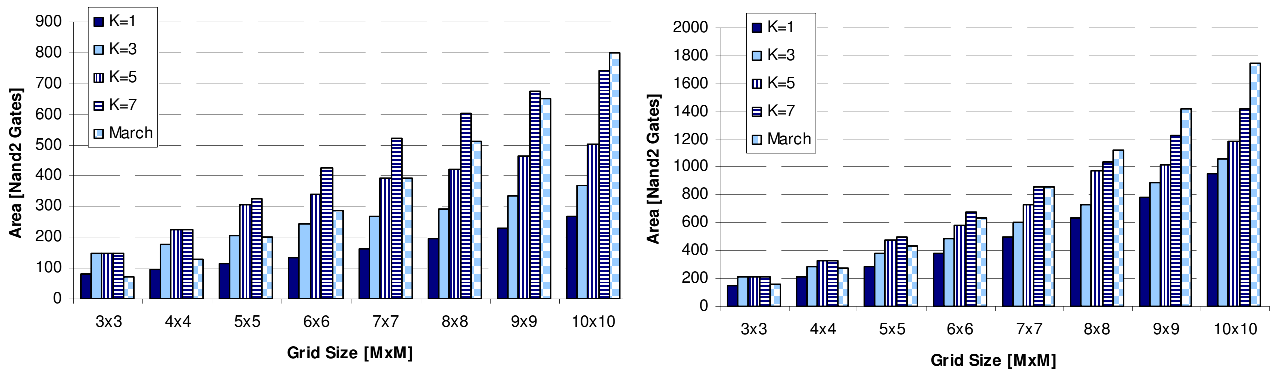


Figure III-14 Area of TSV Interconnect BIST components for an 8x8 TSV array when different K-aggressor orders are considered

For upper-die and lower-die *IBIST* components, the area variation is entirely due to the *TPG*s. For all aggressor orders, *64* MUXes are needed in lower die and *64* RA cells in the upper die, respectively. The *IBIST* cumulated area increases almost linearly from *~800* gates for *K=1* to *~1900* gates for *K=10*, but the TPGs account for less than *25%* of the area. This linear dependence is due to the increase of the number of victim sets with the aggressor order. These results show that the *IBIST* area depends on the chosen aggressor order. For *K=10*, KAF-based tests are identical to *MAF*-based tests. Hence, for *K<10*, the area of the KAF-based *IBIST* is lower (i.e. up to 3× smaller for *K=1*) than for *MAF*-based tests. For aggressor orders up to *K=7,* the results show that the KAF-based *IBIST* area is less than that of conventional marching-based *IBIST* implementations whose upper and lower die modules require *~700* and *~1075* gates, respectively.

In Figure III-15 (a,b), the area of the *IBIST* upper die and lower die components is estimated for TSV arrays of different size and for different aggressor orders *K*. For comparison, the *IBIST* area for TPGs using the marching-'1' pattern generation algorithm are also evaluated.



(a) Lower die BIST area (b) Upper die BIST area

Figure III-15 The KAF-based TSV BIST area of the lower and upper die components for differet TSV array sizes and aggressor orders

For a given aggressor order *K*, the area of *IBIST* components increases with the number of TSVs, as the number of 2:1 MUXes and RA cell increases. The area increase is also attributed to TPGs, as the number of victim sets also increases with the grid size (see Figure III-13). However, for low aggressor orders (i.e. *K=1*), the *IBIST* use the same TPG circuitry and the extra area is entirely due to lower-die MUXes and upper-die RAs.

For a given number of TSVs, increasing the aggressor order leads to larger *IBIST* area, as the TPGs are more complex. Overall, the impact of TPGs on the *IBIST* area becomes significant, as the gap between $K=1$ and $K=7$ increases with the number of TSVs. For larger $M \times M$ TSV bundles, low-order KAF-based *IBIST* implementations occupy up to 2× less area than high order implementations.

Compared to marching '1'-based *IBIST*, the KAF-based strategy requires less area for large grid sizes and low aggressor orders. Moreover, marching-'1' test strategies have another major disadvantage: they can sensitize only open, short and delay faults, but no delay faults due to crosstalk.

3.4.2.2 *Configurable IBIST*

The configurable KAF *TPG*s alleviates some limitations of the simple scheme by choosing different aggressor order K for tests during system lifetime. The 8×8 TSV array has the maximum aggressor order 10 and the aggressor order can range from 1 to K_{MAX}, where $2 \le K_{MAX} \le 10$. In Figure III-16, the impact of the maximal aggressor order K_{MAX} on the *IBIST* area is represented.

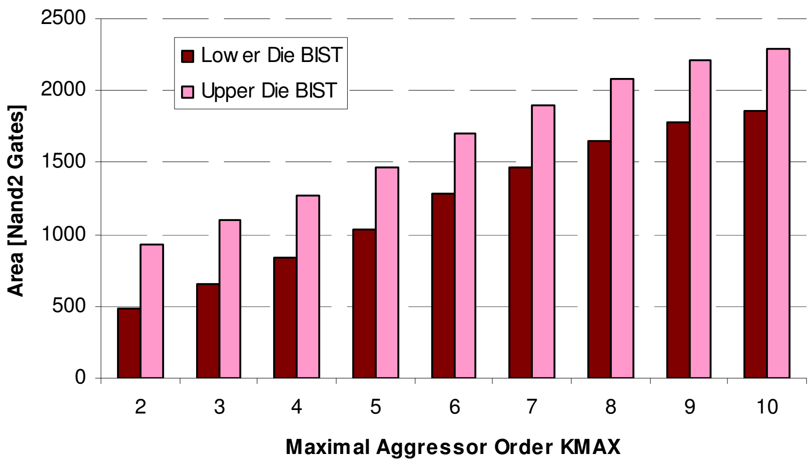
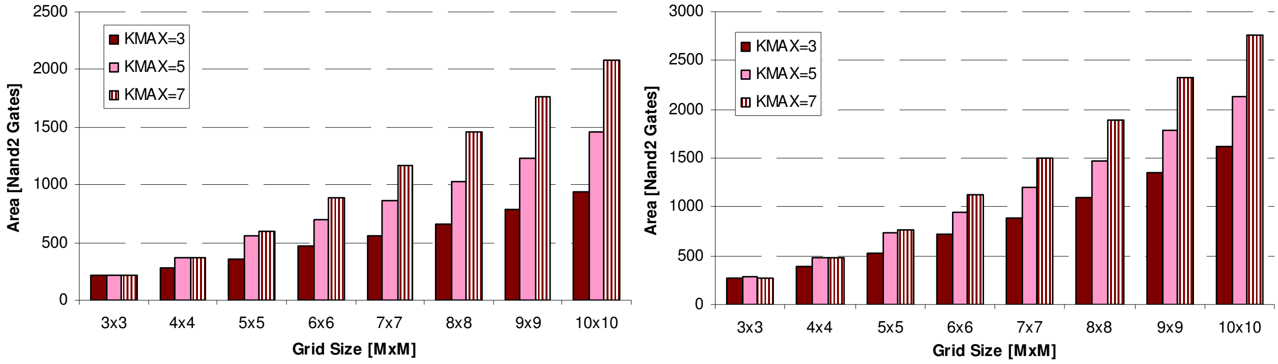


Figure III-16 Area of the configurable KAF-based BIST component for different maximal aggressor orders KMAX

The area overhead increase of configurable *IBIST* is non-negligible. The extra area is entirely due to the *TPG* modules, as the number of *MUX*es and *RA* cells is the same (i.e. 64 for the 8×8 TSV array). Increasing the maximal aggressor order leads to complex TPGs. Hence, the *IBIST* area increases by up ~175% when K_{MAX} increases from 2 to 10. The major contributor to the area overhead is the crossover switch with w_{MAX}-inputs and 64-outputs. In the case of $K_{MAX}=10$, the crossover switch accounts for more than 55% of the *IBIST* area.

In Figure III-17 (a,b), the area of the configurable *IBIST* upper die and lower die components is determined for TSV arrays of different size and for different maximal aggressor orders.



(a) Lower die BIST area (b) Upper die BIST area

Figure III-17 Area of configurable KAF-based BIST components for different maximal aggressor orders and grid sizes

For a given maximal aggressor order K_{MAX}, the *IBIST* area increases with the grid size, as the number of 2:1 MUXes and *RA* cells increases. The increasing complexity of TPGs also increases the overall area, as for each TSV there is a 2:1 MUX that select the aggressor or the victim signal. For a given TSV grid size $M \times M$, the *IBIST* area increase is entirely due to the TPG. For 3×3 and 4×4 TSV bundles (i.e. 9 and 16 TSVs, respectively), there are no variations with K_{MAX}, because the maximum aggressor order is $K_{MAX}=4$ and $K_{MAX}=6$, respectively. It can be noticed that the area difference between the *IBIST* with low (i.e. $K_{MAX}=3$) and high (i.e. $K_{MAX}=7$) aggressor orders increases with the grid (i.e. number of TSVs) size.

Although the KAF-based *IBIST* configurability costs are significant, one of its benefits is the reduction of interconnect test duration during system lifetime. In the case of K_{MAX}=10, the initial tests performed at $K=10$ take 512 cycles. Subsequent tests during system lifetime can be performed at $K=1$, which requires only 16 cycles (i.e. *32* times faster). Hence, if m out of M test phases are performed with $K=1$ instead of $K=10$ then test times are reduced by a total of $m\cdot 496$ cycles. Therefore, this solution becomes attractive in systems that regularly perform TSVs tests. Another possible advantage of this implementation is to increase the aggressor order K of the KAF-based test during system life-time, in order to take into account the TSV aging defects. Finally, this strategy can also be used for calibrating a TSV technology before production. In other words, different TSV configurations are fabricated and an optimal aggressor order is experimentally determined.

3.5 *Conclusion*

Facing the challenges of testing 3D integrated systems, a TSV Interconnect Built-In Self-Test technique (*IBIST*) to cope with opens, shorts and delay faults due to crosstalk is presented. The proposed test strategy uses the novel K^{th}-Aggressor Fault (KAF) model. For a victim TSV, its potential aggressors are divided based on their orders (i.e. *K*-orders), depending on their distance to the victim. Compared to existing delay fault-aware test methodologies, the test times are significantly improved, as several victim TSVs are concurrently tested.

The KAF model is used for implementing the Test Pattern Generator (TPG) of a generic TSV *IBIST* architecture. Experimental evaluations showed that, along with lower test times, the proposed KAF-based *IBIST* occupies up to *three times* less area than *IBISTs* using the maximum aggressor fault model (MAF) or the marching-‘1’/’0’ algorithm. The main challenge on the KAF model is to choose the value of *K* that ensures minimal test time without fault coverage loss. To solve this problem, a configurable TPG implementation allowing TSV tests for different aggressor orders has been proposed. Compared to the original proposal, the configurability costs are high, but the test times can be significantly reduced during system life-time. Increasing the aggressor order K during system lifetime could help detect aging defect. Also, the configuration capabilities may be useful in calibrating a TSV technology before full-scale production.

The reliability and yield remain major challenges of 3D MPSoCs. Although pre-bond testing helps in discarding dies with faulty intra-die components, TSV failures remain the major cause of yield loss. In 3D Networks-on-Chip, this problem is addressed by spare-based repair, serialization and fault tolerant routing in Chapter 4. Transient faults, which affect data bits during link and router traversal, reduce system reliability and may lead to system failures. Thus, *data link* and *network*-level solutions based on error control strategies are also proposed in the following chapter.

Chapter Four

ERROR RESILIENCE IN 3D NETWORKS-ON-CHIP

We have seen that testing the 3D NoC inter-die links can diagnose TSV permanent faults due to manufacturing defects or aging. Without any repair mechanisms, permanent faults affect messages transmitted on the NoC and they could have dramatic effects on the system behavior. Transient faults, which may not be diagnosed during TSV tests, could also affect the integrity of data traversing the NoC. Therefore, error resilience strategies are mandatory in order to ensure correct data transmission. In this chapter, the 3D NoC reliability and yield challenges due to unreliable TSV technologies are addressed at two abstraction layers: data link level and network level. Although single-layer solutions are able to cope with TSV faults, their efficiencies are often counter-balanced by their relatively high costs. In order to cope with these issues, a multi-layer approach, which leverages data link and network solutions, is proposed.

4.1 *Data link error resilience for transient faults*

Error resilience against transients on the inter-die and intra-die links of 3D NoCs is ensured using hardware redundancy and error control schemes. At the transmitter side, the flow control units (*flits*) are encoded before being sent on the physical wires (*PHY*). The received encoded flits are checked for transmission errors. If errors are detected then they are handled by the error recovery mechanism, which is usually implemented by means of error correction or flit retransmission. The flow control signals ensure the NoC correct behavior. Transient faults on these signals are most likely to lead to system failure. Therefore, aggressive protection strategies such as *triple modular redundancy* (*TMR*) are used.

In this section, implementations of the three main classes of error control schemes are presented. *Forward Error Correction* (*FEC*) schemes rely on error correction codes that are capable to correct an arbitrary number of errors on the link. *Automatic Retransmission Query* (*ARQ*) schemes use data retransmission error recovery mechanisms every time transmission errors are detected. Finally, H*ybrid Error Correction and Retransmission* (*HYB*) schemes combine the first two schemes such that if correction is not possible then the flit is retransmitted.

4.1.1 Forward Error Correction

Forward Error Correction (FEC) schemes correct one or more erroneous bits that affect signals on a given link. Error resilience is ensured by means of error correction codes (ECC) [CL06], which are capable of correcting one or more transmission errors, depending on the Hamming distance of every selected code. In Figure IV-1, two routers T_X and R_X connected by a NoC link with *FEC* protection are represented.

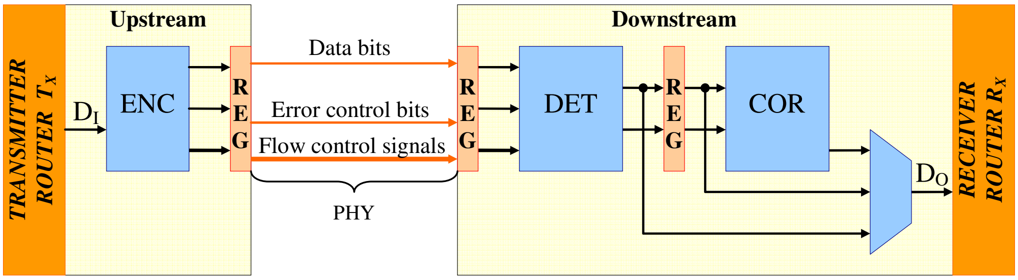


Figure IV-1 Forward Error Correction (FEC) scheme

On the transmitter T_X side (i.e. upstream interface), flits are encoded by the *ENC* module. Data and error-control bits are then transmitted on the physical wires (*PHY*). The error detection module *DET* on the receiver R_X side, checks the received signals for transmission errors. The error correction module *COR* corrects detected errors before flits are sent to the subsequent stages. Depending on circuit timing, one or more retiming stages could be necessary between T_X and R_X. These stages could be inserted between the encoder and the *PHY*, between the *PHY* and the *DET* module, or between the *DET* and the *COR* modules.

In the worst case, data encoding, transmission on *PHY*, error detection and correction take one clock cycle each, and the three retiming stages are considered. If no transmission errors are detected then the correction module is bypassed and flits traverse the link in two cycles. If errors are detected then an extra cycle is required for error correction. During this correction cycle a new data could arrive. Therefore, in order to prevent data conflicts on D_O, a *Finite State Machine* (*FSM*) controls the downstream interface. Once an error is detected, the correction stage bypass mechanism is disabled. Flits go through the correction stage and link latency increases to three cycles. In other words, once an error is detected, all subsequent flits are delayed for one cycle, even if no errors are detected. The correction bypass remains disabled until all the flits of the current burst are transmitted. In Figure IV-2, the time diagrams of flits traversing the *FEC*-protected link are presented.

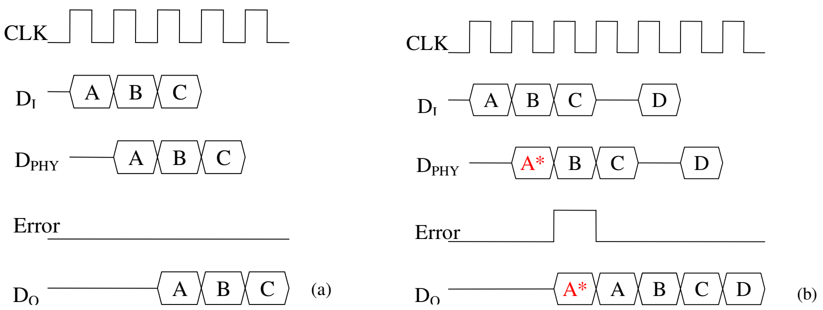


Figure IV-2 Time diagram of flit arrival at the receiver router when there are no faults (a), when errors are detected for flit A (b)

When A is affected by transient errors during *PHY* traversal ($A*$ on D_{PHY} in Figure IV-2(b)), the errors are detected in the second cycle. In this case, flit $A*$ is discarded at the downstream interface and it is not forwarded to the receiver R_X. An extra cycle is needed for error correction. Therefore, flit A will arrive error-free at the receiver router after three cycles. Flits B and C follow A in a pipelined way such that the correction delay on A also affects their arrival time. Since C is the last flit in the burst, after it traverses the detection stage, the downstream *FEC* interface returns to normal operation (i.e. correction bypass enable). Flit D, which is the first flit of a new burst, traverses the link in two cycles if no errors are detected.

The link reliability (i.e. probability that flits traverse the link without any faults) depends on the correction capabilities of the coding scheme. Multiple error correction coding schemes like Low-Density Parity Check (*LDPC*) and Reed-Solomon (*RS*) are too complex and prohibitively expensive for NoC links. Therefore, most implementations use different forms of Hamming Single Error Correction (SEC) codes. For such codes, it is important to know the codeword size that can be reliably transmitted on the *PHY*.

Let us assume the single wire error rate ε_{wire} and a target flit error rate ε_T, which represent the probability that an n bits link is affected by an uncorrectable error. The goal is to determine the link size n such that Hamming SEC protected flits reliably traverse it (i.e. the probability of double errors is less than ε_T). For Hamming SEC codes, the number of error control bits m is the smallest number that satisfies $2^m \geq n+m$. Hence, n is determined such that the probability of an uncorrectable error (i.e. multiple error $1-\varepsilon_0-\varepsilon_1=\varepsilon_2+\varepsilon_3+\ldots+\varepsilon_{n+m}$, where ε_i represents the probability that i out of $n+m$ bits are affected by faults) is less than ε_T. In Figure IV-3, the number of data bits n that can be reliable transmitted is represented for different single TSV fault rates and target error rates.

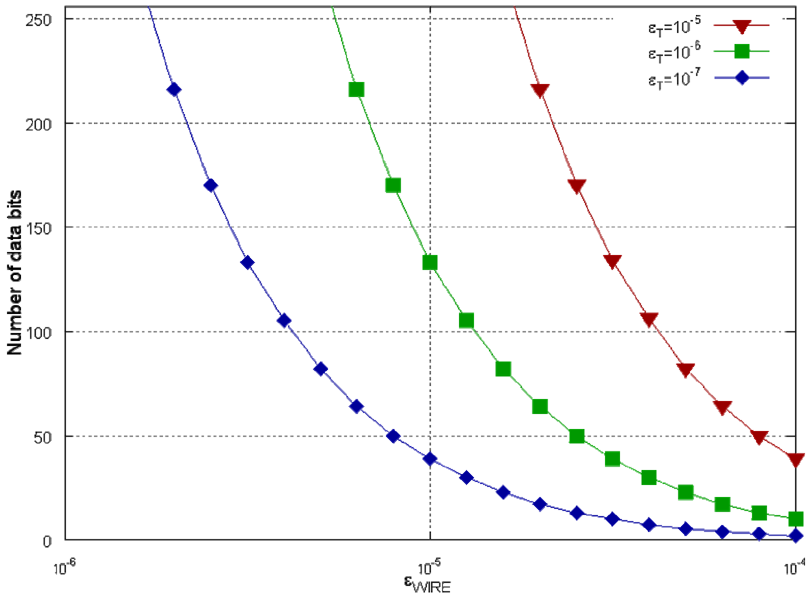


Figure IV-3 Data size for reliable transmission using FEC

The results above show that the number of data bits that can be reliably transmitted on *PHY* decreases if the single wire error rate ε_{wire} increases for a given flit error rate target ε_T. For a constant error rate ε_{wire}, higher reliability targets can be achieved only when fewer data bits are transmitted. Hence, in the case of error-prone communication channels, the maximal number of transmitted data bits must be reduced in order to achieve the reliability targets. For example, it is not possible to sent more than *48* data bits with an error rate below 10^{-7} on a *PHY* with $\varepsilon_{wire} \geq 10^{-5}$.

Data bits protection during transmissions can be improved by implementing codes with multiple error correction capabilities. When multiple error correction is necessary, SEC codes are interleaved, composed or modified in order to correct particular multiple fault patterns. In interleaved SEC codes data bits are split in two or more disjoint groups and each of these groups is encoded using a SEC code. In general, having more groups increases the correction capabilities of the code: when data is split in *g* groups, all multiple error patters of up to *g* errors can be corrected if the errors are distributed such that there is at most one error per group. The correction capabilities of interleaved Hamming codes represent the probability that there is at most one error per group. In Figure IV-4, the correction probabilities are represented for 32 data bits split in one, two, three and four groups.

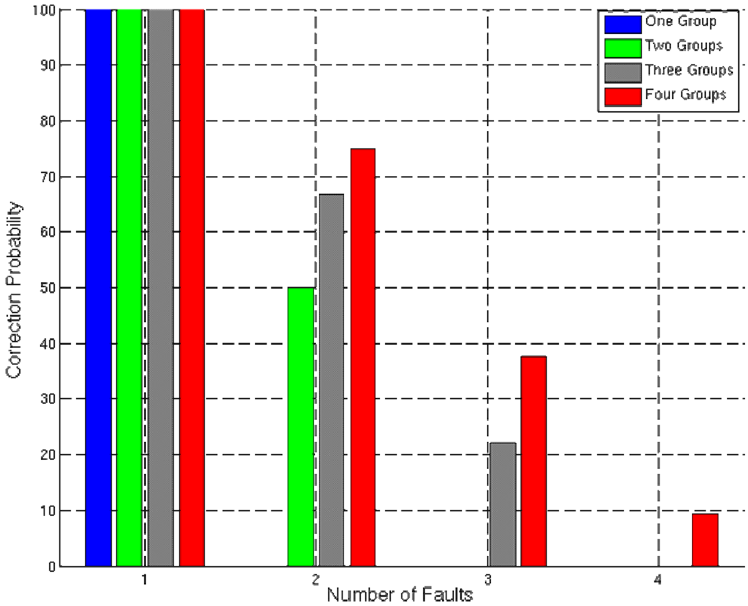


Figure IV-4 Correction probabilities of interleaved Hamming SEC codes for 32 data bits

As expected, the correction capabilities increase with the number of groups. For a uniform error distribution, *~50%* of all double errors are correctable when two groups are used. However, when there are four groups, this probability increases to *~75%*, as it is less likely to have two errors in the same group. Data interleaving affects the codeword size and the number of wires needed for data transmission. For example, *32* data bits can be encoded on *38* code bits using a single group Hamming SEC, *42* code bits for two groups and *48* for four interleaved groups. This has a non-negligible impact on the number of wires / link and the overall TSV footprint.

4.1.2 ***Automatic Retransmission Query***

Automatic Retransmission Query (*ARQ*) schemes rely on data retransmission to recover after transmission error detection. In the upstream link interface, flits are encoded and stored in dedicated retransmission buffers (RT_{FIFO}) before being sent on the *PHY*. Transmission errors are detected in the downstream interface by the error detection module *DET*. If no errors are detected then flits are forwarded to the receiver R_X. If errors are detected (i.e. *Error='1'*) then the faulty flit is dropped and a retransmission request is made. The *ARQ* mechanism represented in Figure IV-5 is also referred as *go-back-N* retransmission, where *N* refers to the number of cycles between the moment when the error was detected and the retransmitted data is received.

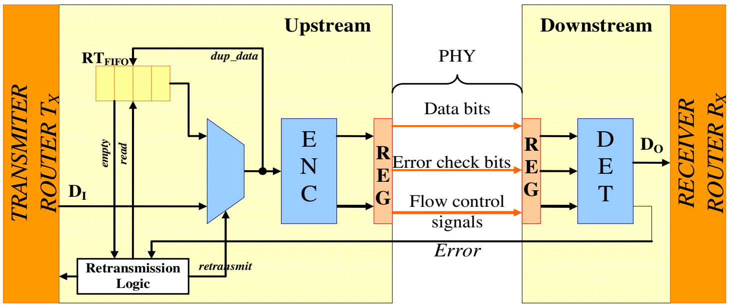


Figure IV-5 Go-back-*N* Automatic Retransmission Query (ARQ) scheme

Upon the reception of the detect error signal in the upstream interface, the *Retransmission Logic* stalls T_X and starts sending flits from the retransmission FIFO. After all the flits initially stored in RT_{FIFO} have been correctly received, the *Retransmission Logic* enables new flit transmissions from T_X. Depending on the circuit timing, fault-free data transmission can take up to two cycles, as two retiming stages are considered between the T_X and R_X routers.

The RT_{FIFO} is implemented as a barrel shift buffer whose size is determined by the number of cycles between the moment when the flit is sent and the moment when transmissions from T_X are disabled by the retransmission request for the initial faulty flit. In Figure IV-6, the time diagram of the *ARQ*-protected link is represented for a retransmitted flit *A*.

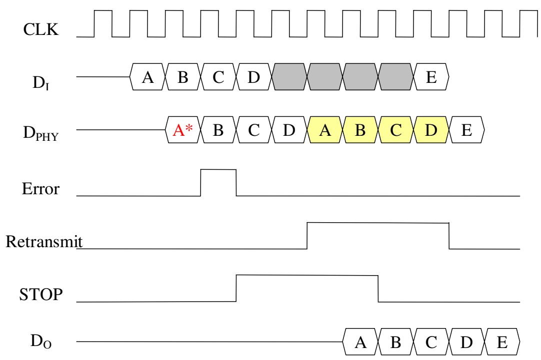


Figure IV-6 Time diagram of go-back-N retransmission when errors are detected

After the error is detected for flit *A*, the upstream *Retransmission Logic* disables transmissions from T_X. However, by the time T_X is stopped and retransmission starts, thee flits B, C, and D, could have been sent. These flits will arrive at the downstream interface where they will be discarded. While T_X is stalled, flits A, B, C and D are resent from the retransmission FIFO. If these retransmitted flits arrive fault-free then they are forwarded to the receiver router (D_O). If errors are detected for the retransmitted flits then all subsequent retransmitted flits are dropped and a new retransmission phase starts. The retransmission buffer must have at least *four* positions, as the retransmission penalty is four cycles. If no retransmission request is received within four cycles after a flit is sent, then it is assumed that it was correctly received and the flit is discarded from RT_{FIFO}.

The error detection capabilities of the coding scheme are very important in ensuring reliable communication. Higher error detection capabilities reduce the probability of flits leaving the link affected by errors. In *ARQ*, error detection is done using error detection codes like parity, interleaved parity (i.e. data bits are split in groups and a parity bit is added for each group) or Cyclic-Redundancy-Check (CRC) codes. Despite the improved error detection capabilities of interleaved parity codes, increasing the number of groups has an impact on the number of wires used for parity bits transmission. For m interleaved groups there are m extra wires used for parity bits. However, the main limitation of interleaved parity encoding is that it cannot detect all multiple error patterns. In order to improve the error detection probability, parallel CRC codes are implemented. In general, CRC codes with a generator polynomial $G(X)$ of degree p can detect all burst errors of length less than p.

In the *ARQ* scheme, if errors are detected after a predetermined number of consecutive retransmission cycles then it is very likely that errors are due to permanent faults. The number of allowed retransmissions depends on the probability of transients in successive transmission cycles. Since this probability is often very low, a single retransmission cycle can be considered. If a flit is not correctly received the first time, then it is very likely that it will be received fault-free the second time. Errors detected for retransmitted flits are most likely caused by a permanent fault. In this case, specific error recovery strategies such as spare-based and serialization-based repair could be used.

4.1.3 Hybrid Error Correction with Retransmission

The *Hybrid Error Correction with Retransmission* (*HYB*) scheme combines the *FEC* and *ARQ* solutions presented above. In the upstream interface, flits are encoded and simultaneously copied in the retransmission buffer RT_{FIFO} before being sent on the *PHY*. On the receiver's side, if the detected error cannot be corrected then a retransmission request for the flit is sent. In Figure IV-7, the *HYB* error resilience scheme is presented.

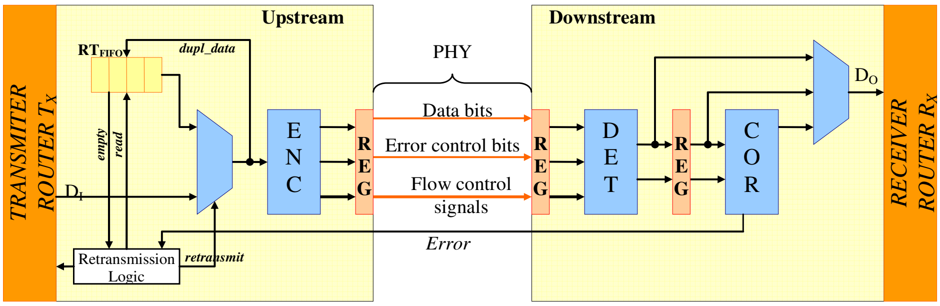


Figure IV-7 Hybrid error correction and retrasmission scheme for NoC links

In a fault-free transmission, flits require two cycles to traverse the link, as two retiming stages are inserted. When an error is detected and corrected, the link latency increases to three cycles. The correction block bypass mechanism is controlled by an *FSM* similar to the one used in the *FEC* scheme. Hence, after a fault correction, the latency remains three cycles until the burst ends or a retransmission request is done. If an error is detected, but cannot be corrected, then the latency increases, as the flits go through the retransmission mechanism similar to that of *ARQ*. In this case, the RT_{FIFO} size has an extra position, as the correction circuitry delay is included in the retransmission roundtrip delay. In Figure IV-8, the time diagram of different faulty flits transmissions are represented.

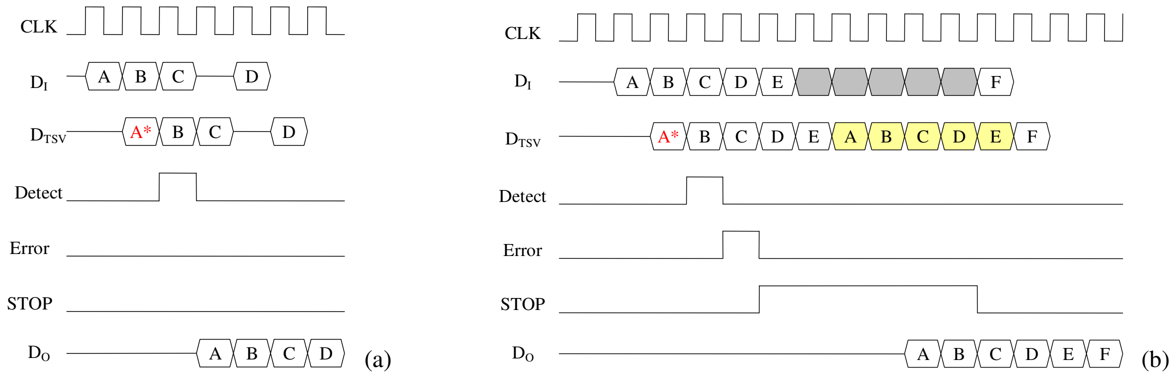


Figure IV-8 Time digram for HYB go-back-N scheme

In the case of a correctable transmission error shown in Figure IV-8 (a), the protected link behaves like the *FEC*-protected one. The delay for the flits *A*, *B*, *C* is three cycles and the delay of flit *D*, which is sent in a different burst, is two cycles. In the case of a detected un-correctable error represented in Figure IV-8 (b), flits *B*, *C*, *D* and *E* are sent before the *STOP* signal disables T_X. These flits are discarded at the downstream interface and they are retransmitted from RT_{FIFO} after flit *A*.

Because an extra cycle is necessary for error correction, the RT_{FIFO} is implemented as a barrel shift buffer with at least *five* positions. If the transmitter does not receive a retransmission request within five cycles after the flit is sent then it assumes that it was correctly received and it is discarded from RT_{FIFO}. After a retransmission request, flits are sent from the RT_{FIFO} until they are correctly received. In the case of uncorrectable errors detected after a predetermined number of retransmission cycles, higher-level error recovery mechanisms (e.g. packet dropping, retransmission) are necessary. In the uncorrelated fault model, transients are unlikely to affect consecutive retransmission cycles. Therefore, the *Retransmission Logic* module is greatly simplified by allowing a single retransmission for each flit.

Unlike *FEC* schemes where only error correction is necessary, in *HYB* schemes both error correction and detection capabilities are used. Hamming SEC codes cannot be used since there is the syndrome aliasing problem (i.e. double errors and single errors have the same error syndrome) that cause a miscorrection in the case of double errors. Thus, Hamming SEC codes extended with a parity bit such that it has single error correction and double error detection (SECDED) capabilities are used.

Since the link reliability depends on the code correction and detection capabilities, it is important to determine the maximal number of data bits that can be reliably sent on *PHY*. Let us consider an *n* bits link with a single transient fault rate ε_{wire}. The objective is to determine the data size *n* that can be transmitted such that the probability of having an undetectable error is less than a flit reliability target ε_T. Unlike in Figure IV-4, *n* is determined such that there are at least *three* out of $n+m$ bits are affected by errors (i.e. $\varepsilon_3+\varepsilon_4+\ldots+\varepsilon_{n+m}$), where *m* is the number of error control bits. If a double error is detected then a retransmission request is made and the probability of having double errors in consecutive transmission cycles is very low (i.e. less than 10^{-20} for the uncorrelated fault model). In Figure IV-9, the number of data bits that can be reliably transmitted on TSV-based communication channels using Hamming SECDED encoding is represented for different single TSV fault rates and target error rates.

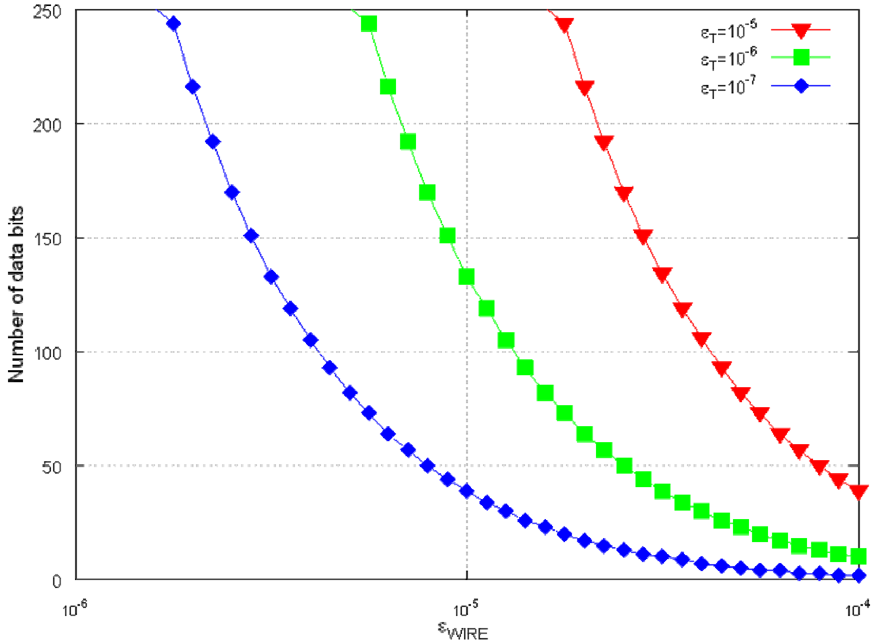


Figure IV-9 Data size for reliable transmission using HYB

Compared to the Hamming SEC codes used in FEC, more data bits can be reliably transmitted for the same error rate ε_{wire} and flit error target ε_T. While more than *128* data bits, cannot be sent on FEC-protected links with error rates ε_{wire} >10^{-5} and targets less than 10^{-7}, *HYB*-protected links can reliability send the data bits even for a flit error rate below 10^{-9}. However, as shown in Figure IV-8, the link latency increases significantly in case of a retransmission.

Since the retransmission penalty is non-negligible, improving error correction capabilities reduce the probably of a retransmission. Similarly to *FEC*, Hamming SECDED codes are interleaved such that multiple errors can be corrected.

4.1.4 Link protection strategies in 3D NoCs

3D NoCs consist of many inter-die and intra-die links that are implemented in different interconnect technologies. The error resilient links presented in the previous sections protect flits against transmission errors on the *PHY*. In a 3D network, each flit traverses many links and routers on its source-destination path. Maximal transmission reliability is achieved by protecting *all* links along the path. Note that it is not necessary to implement the same error resilience scheme on all links. This heterogeneous protection strategy is applied for networks containing links with different *PHYs* and error rates. Despite the reliability benefits, this protection strategy comes with high performance penalties, as the latency of each link increases by at least *one* cycles, and non-negligible costs.

An alternative is to trade reliability for performance, area and power, by protecting some links of the NoC (i.e. *selective* protection). This way, source-destination paths comprise both protected and un-protected links. In the case of selective link protection, if the path reliability constraints are satisfied then the performance penalties and overheads are reduced.

For 3D NoCs, the selective protection strategy consists in protecting only inter-die links. This way, transient errors that occur on the inter-die *PHY* are mitigated, resulting in some path / network reliability degradation. In Figure IV-10, the inter-die link protection strategy is represented for regular and quasi-regular 3D NoC topologies.

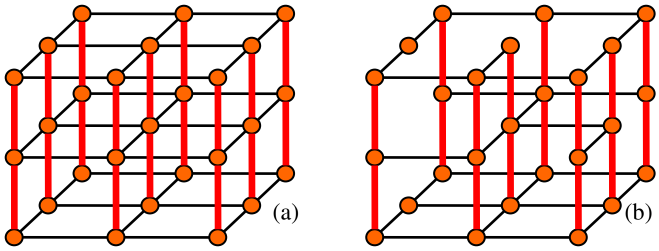


Figure IV-10 3D NoCs with regular (a) and quasi-regular (b) topologies using inter-die protected links

Because only inter-die links are protected, transients on intra-die links cumulate along the source-destination path, reducing the network reliability. If the selective inter-die protected network reliability is less than the reliability target then the error resilience strategies may be implemented also for intra-die links.

The error control schemes presented above target only transient faults and they often fail if wires are affected by permanent faults. At data link level solutions like spare-based repair and serialization can be used for masking such faults. In the following section, different data link strategies for TSV repair are presented.

4.2 *Data Link error resilience for permanent faults*

Data link-level TSV repair techniques must deal with TSV permanent faults due to manufacturing and aging defects. In 3D NoC links, permanent TSV faults due to manufacturing defects are mitigated using spares (i.e. *TSV Spare-and-Replace*) and serialization (i.e. *Configurable fault-tolerant Serial Links*) techniques. Using the off-chip resources of external testers, TSVs are tested for structural faults (e.g. open, short) and the repair signals are computed. For systems comprising hundreds and thousands of TSVs, ensuring high reparability using spare-based repair could require more TSV / chip than available on-chip. Therefore, at the expense of serialization / deserialization circuitry and reduced inter-die link latency and throughput, high TSV yield can be ensured without using spares.

In complex 3D chips having thousands of TSVs, the off-chip test and repair processes take a long time. Moreover, permanent TSV faults due to aging and wear-out, which are not covered by off-chip test and repair strategies, are also likely to occur. Therefore, on-chip self-test and self-repair strategies may become necessary and the *Interconnect Built-In Self-Repair and Adaptive Serialization (IBIRAS)* strategy is jointly used with the *IBIST* technique presented in Chapter 3.

4.2.1 TSV Spare-and-Replace (TSV-SnR)

Spare-and-replace (*SnR*) is a fault-tolerant strategy based on hardware redundancy that consists in replacing faulty components with functional spares. In 3D integrated systems, the TSV manufacturing yield is improved by allocating some redundant TSVs that are used for replacing faulty regular TSVs. In this section, an implementation of the *TSV-SnR* scheme is presented. A repair costs reduction strategy based on TSV grouping is also presented.

4.2.1.1 TSV-SnR Architecture

After 3D chip packaging and final tests, faulty TSVs are identified using an interconnect test strategy (e.g. *Boundary-Scan*). Using an on-chip repair fabric (i.e. *crossover switch*), faulty TSVs are replaced with functional spares. In order to reduce the timing overhead of signal rerouting on functional spares, it is considered that regular and redundant TSVs are in the same bundle. In Figure IV-11, the *TSV-SnR* repair modules are represented for a bundle with n regular wires and r spares.

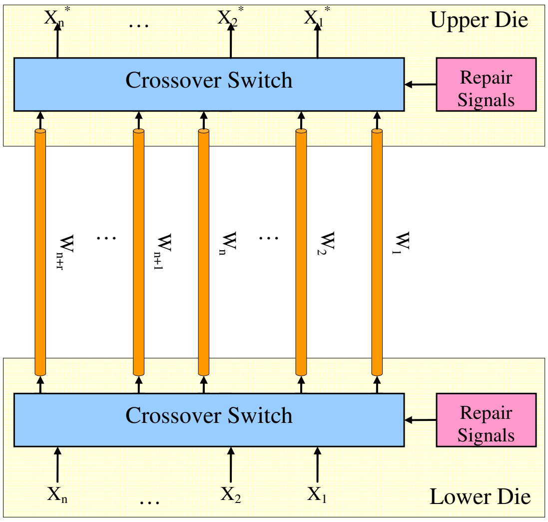


Figure IV-11 Repair fabric for n regular TSVs and r spares

The n signals X_1-X_n propagate from the lower to the upper die through the repair fabric. Crossover switches in the lower die map the input signals $X1$-Xn to n fault-free TSVs in the bundle $W1$-$Wn+r$. In the upper die, the crossover switch performs the reverse operation, remapping the received signals on W_1-$Wn+r$, to their original positions $X1^*$-Xn^*.

The crossover switches are implemented as arrays of switching elements (e.g. pass-transistors or tri-state buffers) or one-hot MUXes. The crossover switch control signals (i.e. repair signals) indicate on which wires signals are mapped. Let us consider that each regular TSV can be replaced by any functional spare. In Figure IV-12, the pass-transistor implementation of the crossover switch is shown.

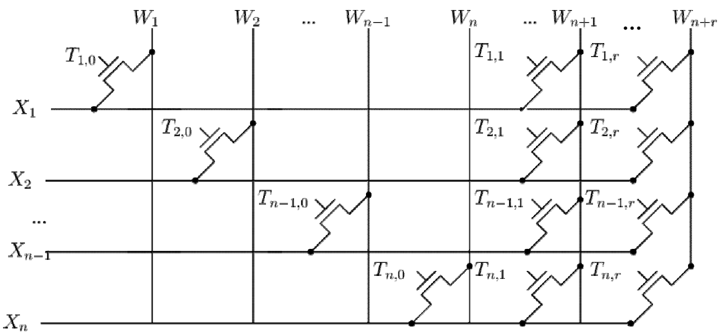


Figure IV-12 Spare-and-Replace crossover switch with n inputs and $n+r$ outputs

It can be seen that each signal X_i can be mapped on its initial position W_i and also on any of the spares W_{n+1}-W_{n+r}. If all regular TSVs are fault-free, each signal X_i is transmitted on W_i. The control (or repair) signals $T_{i,j}$ indicate whether the input signal X_i is mapped on wire W_j. There is one repair signal for each switching element and the $n{\cdot}(r+1)$ signals are stored in the *one-time-programmable* (*OTP*) memory, which is programmed using the scan chain after the repair signals have been determined off-chip. For n regular and r

spare TSVs, the lower die crossover switch has n inputs and $n+r$ outputs. The upper die crossover switch is symmetric, as data signals received on the $n+r$ TSVs are remapped on their original positions.

The crossover switch architecture shown in Figure IV-12 has the disadvantage that there are more drivers for the spare TSVs. Moreover, physical distribution within the TSV bundle could result in different signal propagation delay. Therefore, balanced crossbar implementation similar to those proposed in [GIS06a] can be used in order to minimize the impact of signal rerouting on propagation delays.

The *TSV-SnR* costs are given by the spare TSV footprint σ_{TSV}, crossover switch area $\sigma_{\times\text{-}over}$ and the *OTP* memory σ_{OTP} necessary to store the repair signals $T_{i,j}$. Therefore, a trade-off between the interconnect footprint and circuitry area may reduce the costs, without affecting the interconnect yield. A possible cost optimization strategy based on TSV grouping is presented in the following.

4.2.1.2 Optimization of TSV-SnR costs

Partitioning the regular TSVs in two or more groups and allocating spares for each group may reduce the crossover switch complexity. Let us assume that the n regular TSVs and r spares are partitioned in g groups of sizes: $n_1 \ldots n_g$, and $r_1 \ldots r_g$. In this case, the crossover switch is decomposed in g sub-switches with n_i inputs and n_i+r_i outputs, which is less complex, as $r_1 \ldots r_g < r$.

The challenge of grouped-based *SnR* is finding the spare configuration that achieves the yield target Y_T with minimal costs. To address this issue, let us consider a TSV bundle with n regular TSVs that can be split in g groups, where $g \leq n$. Without loss of generality, it is considered that each TSV has an Y_{TSV} yield, and faults are uniformly distributed. The algorithm used for finding the optimal *TSV-SnR* configuration consists in successive estimation of costs for different group sizes. In Figure IV-13, the algorithm pseudo-code is presented.

```
01: for g=1…n
02:    [n_1,…,n_g] ≈ ⌊n / g⌋;
03:    [r_1,…,r_g] = 0;
04:    Y = Y_TSV^n;
05:    while (Y < Y_T)
06:       i = ((i+1) mod g) + 1;
07:       r_i = r_i + 1;
08:       Y =Yield(n_1,…, n_g; r_1, …,r_g)
09:    end;
10:    area = estimate_area(n_1,…,n_g; r_1,…,r_g);
11:    if (area < smallest_area) then
12:       store(n_1,…,n_g; r_1,…,r_g);
13:    end;
14: end;
```

Figure IV-13 Optimal Spare-and-Replace configuration process

For a given group size g, the regular TSVs are split in g quasi-equal groups $n_1 \ldots n_g$. If correlated faults are considered then the partitioning strategy in line *02* is more complex. For example, if faults are more likely on the TSV bundle boundary then groups containing TSVs at the boundary would be smaller. In lines *05-09*, given a TSV partitioning $[n_1,\ldots,n_g]$, the number of spares is increased for one group at a time until the target yield is achieved. In line *08* the interconnect yield is evaluated using the *Yield()* function for the configuration with *g* groups. The *estimate_area()* function returns the repair fabric area (i.e. area of crossover switches, storage elements for repair signals and spare TSV footprint for the targeted technologies). If the current configuration has smaller area than previous ones then it is stored. At the end of this process, the *Spare-and-Replace* configuration with minimal area for the targeted yield is returned.

When the costs of adding spares are acceptable, the *TSV-SnR* technique can be used. Unfortunately, TSVs are an expensive resource to have and it is not always possible to allocate enough spares such that high yield targets are achieved. For the inter-die links of 3D NoCs, an alternative solution to *TSV-SnR* is fault-tolerant serialization: flits transmission in two or more cycles using functional TSVs.

4.2.2 Configurable Serial Fault-Tolerant Links (CSL)

The *Configurable Fault-Tolerant Serial Link* (*CSL*) methodology ensures high TSV yield without relying only on spares. In 3D NoCs, this technique is used for repairing TSV permanent faults due to manufacturing defects. *CSL* relies on serialization when there are not enough fault-free spares to replace faulty regular TSVs. In this section, the proposed *CSL* architecture and repair process is described along with a cost reduction strategy based on regular TSV grouping.

4.2.2.1 CSL Architecture

In 3D NoCs, let us consider inter-die links with N regular and R spare TSVs used for data transmission. Inter-die links are functional if at least M_{MIN} out of $N+R$ TSVs are fault-free. Thus, the worst-case number of cycles required for serial transmission is $K_{MAX}=\lceil N/M_{MIN} \rceil$. The *CSL* configuration is done after interconnect tests using the diagnosis vector which identifies M functional TSVs. If $M \geq N$ then there are enough functional spares such that all faulty regular TSVs are repaired and data bits are sent in a single cycle. If $M < N$ then groups having up to M data bits are sequentially sent on functional TSVs. The upstream register uses N flip-flops to store the data bits that will be sent in groups of M. In the downstream interface, the received data signals are remapped on their original positions. These signals are loaded in M out of N flip-flops of the downstream register that recreates the original data. In this case, N data bits transmission takes $K=\lceil N/M \rceil$ cycles. In Figure IV-14, the *CSL* upstream and downstream interfaces are connected to the transmitter T_X and the receiver R_X router interfaces.

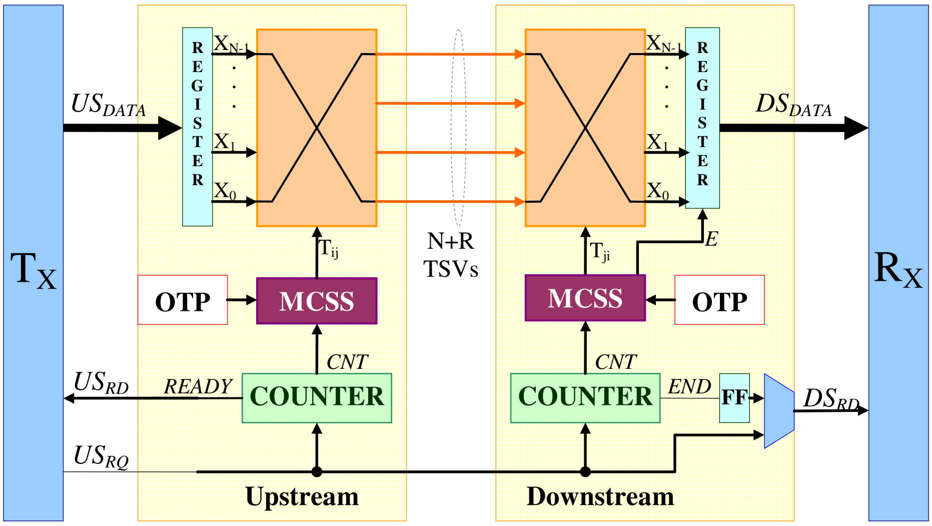


Figure IV-14 N-bits Configurable fault-tolerant Serial Link (CSL)

The transmitter T_X is allowed to send data (US_{DATA}) only when the upstream interface is ready (US_{RD} = '1'). If data is not serialized then the upstream and downstream registers are bypassed, and transmission request are directly passed to the receiver (DS_{RD}=US_{RQ}). In serialization mode, for each transmission request (US_{RQ}='1') the data bits are loaded in the upstream register. Then, the upstream interface disables the link (US_{RD} = '0') until the current transmission ends (i.e. for K cycles). Once the N data bits have been loaded in the downstream register, the downstream interface signals the receiver R_X that the data DS_{DATA} is ready (DS_{RD}='1').

The crossover switches map data signals on and from functional TSVs under the control of the *Matrix Control Signal Selection* (*MCSS*) modules. The crossover switches are implemented as matrix arrays of switching elements like tri-state buffers and pass-transistors, or as one-hot MUXes. Starting from the most significant position, each data bit X_i is mapped on its initial TSV W_{i+R}. As *CSLs* tolerate up to N+R-M_{MIN} faults, when regular wire W_{i+R} is faulty, it can be replaced by one of its rightmost N+R-M_{MIN} neighboring wires: W_{i+R-1}, W_{i+R-2},…,$W_{i-N+MMIN}$. If there are not enough wires to the right (i.e. *W0* is the last wire in the bundle) then wires are reallocated starting from the most significant position *WN+R-1*.

The set of crossover switch control signals $T = (T_{ij})$ indicates on which functional TSVs data signals are mapped. In the upstream interface, the control signal T_{ij} is active when X_i is mapped on W_{i+R-j}, where $0 \leq i < N$ and $0 \leq j \leq N+R-M_{MIN}$. The upstream and downstream crossover switches and their control signals are transposed, as data bits mapped on different wires must be remapped on their original positions. When data is serialized, the control signals T command the crossover switches to map at most M data bits on the functional TSVs. This process is done for each transmission cycle and $K=\lceil N/M \rceil$ sets of control signals T_0, T_1,…, T_{K-1} must be used for sending the N bits message. The current state of the counter (*CNT*) is used by the Matrix Control Signal Selection (*MCSS*) module to select the corresponding set T of control signals. The pass-transistor implementation of the upstream crossover switch for four bits links with R=*1* spare is represented in Figure IV-15. This link is functional if at least M_{MIN}=*2* of its wires are fault free.

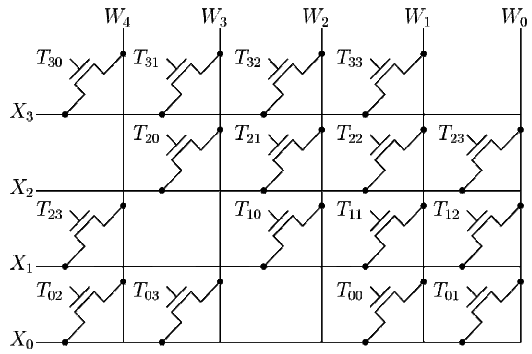


Figure IV-15 4-inputs and 5-outputs upstream crossover switch with the corresponding control signals

For the 4-bits link with one spare, each data bit can be mapped on four wires. As mentioned before, these wires are allocated starting with the most significant position. Hence, data bit X_3 can be mapped on any wire W_4-W_1, and X_2 is mapped in W_3-W_0. It can be noticed that for X_1 there are no remaining wires to the right. Hence, a wrap-around is used such that X_1 can be mapped on W_2, W_1, W_0, and W_4. Similarly, X_0 can be mapped on W_1, W_0, W_4, and W_3.

When data serialization is enabled, the counters in the link interfaces are configured to count the K transmission cycles. When transmission starts (US_{RQ}=*'1'*), they are initialized and the link status is determined by their status signals *READY* and *END*. The upstream signal disables transmissions from TX when it counts the transmission cycles (i.e. *READY='0'*). The *END* signal is activated when the counting sequence has finished. In the downstream interface, this signal is delayed one clock cycle and it indicates that serial transmission finished and data is ready ($DS_{RD}=END^{clk+1}$). If there are clock constraints that impose one or more register stages between the two interfaces then the downstream counter initialization signal must be delayed by the same number of clock cycles as the pipeline length.

The CSL serialization cycles K, repair signals T_0, T_1 …T_{K-1}, and downstream register enable signals E_0, E_1…E_{K-1} are determined off-chip using the interconnect test diagnosis vector. These signals are stored in a one-time-programmable (*OTP*) memory at the CSL upstream and downstream interfaces. In the following, the off-chip repair signal computational process is presented.

4.2.2.2 *Off-chip repair signal computation*

A total of $K=\lceil N/M \rceil$ different sets T of control signals are used when links are configured to serialize data. Therefore, a maximum of K_{MAX} sets of $N\cdot(2\cdot N+R-M_{MIN}+1)$ control signals are stored in each CSL interface. The functional blocks used to compute the repair signals off-chip are represented in Figure IV-16.

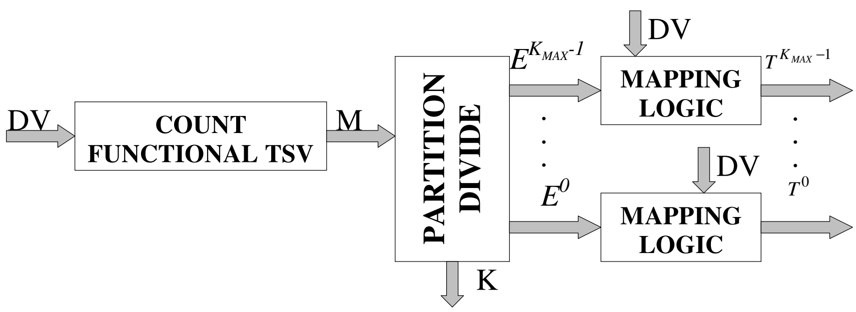


Figure IV-16 Computational blocks used to determine the number of transmission cycle K, the matrix control signals T, and the downstream register enable signals E

First, the number of functional wires M is determined by counting *0*s in the diagnosis vector DV. If there are not enough functional wires then the link is faulty. If there are at least M_{MIN} functional wires then the number of transmission cycles K and the flip-flop enable signals E are computed for each transmission cycle by the *PARTITION/DIVIDE* block. The downstream register enable signals $E0,\ldots,EK\text{-}1$ indicate on which positions data signals are transmitted. For example, if the *ith* position on Ek is*'1'* then the *ith* data bit Xi is transmitted in the *kth* transmission cycle. The crossover switch control signals T are determined by the *MAPPING LOGIC* block using the register enable signals of each transmission cycle and the diagnosis vector.

If $M \geq N$ then data bits $X_{N-1}, \ldots, X_0$ are sent in a single cycle $K = 1$ and the flip-flop enable signals E_0 may have any values, as the upstream and downstream registers are bypassed. If $M<N$ then transmission in a single cycle is not possible and data bits $X_{N-1},\ldots,X_0$ are split in K groups of size at most M. A simple way to determine which data bits are sent is to consider groups of M starting from the most significant ones. Thus, the enable signals of the most significant M FFs are active in the first cycle, then the following M signals and so on. In general, data bits $X_{N-kM-1},\ldots,X_{N-(k+1)M}$, are sent in the k^{th} transmission cycle, where $0 \leq k < K$. If M does not divide N then data bits $X_{(N \bmod M)-1},\ldots,X_0$ are sent in the last transmission cycle.

The crossover switch control signals T and the enable signals E are determined from the diagnosis vector for each transmission cycle. When X_i is sent in the k transmission cycle, the downstream register i^{th} *FF* is enabled, i.e. E^k_i =*'1'*. Data bit X_i is mapped on the functional TSV W_{i+r-j} (i.e. DV_{i+r-j} = *'0'*) if it is not used by any data bits $X_{n-1},\ldots,X_{i+1}$ sent in the same cycle. With these conditions, the recursive expression of T_{ij} in the k$^{\text{th}}$ transmission cycles is given in equation (1).

$$
\begin{aligned}
&T_{i,j}^{K} = E_i^{\mathrm{K}} \cdot \overline{DV_{i+r-j}} \cdot \overline{T_{i,0}^{K} + \ldots + T_{i,j-1}^{K}} \cdot \overline{T_{i+1,j+1}^{K} + \ldots + T_{I_{MAX},(j+i-I_{MAX}) \bmod M}^{K}} \\
&i = \overline{N-2:0},\ j = \overline{0:N+R-1+M_{MIN}} \\
&I_{MAX} = MIN(N-1, i+M_{MIN}-j-1)
\end{aligned}
\tag{1}
$$

Where I_{MAX} identifies the most significant data bit that can use wire W_{i+R-j}. As data bits are mapped starting from the most significant bit X_{N-1}, the initial conditions of equation (2) are:

$$
\begin{aligned}
&T_{N-1,N+R-1}^{K} = E_{N-1}^{\mathrm{K}} \cdot \overline{\mathrm{DV}_{\mathrm{N+R-1}}} \\
&T_{N-1,N+R-2}^{K} = E_{N-1}^{\mathrm{K}} \cdot \mathrm{DV}_{\mathrm{N+R-1}} \cdot \overline{\mathrm{DV}_{\mathrm{N+R-2}}} \\
&T_{N-1,N+R-j}^{K} = E_{N-1}^{\mathrm{K}} \cdot \mathrm{DV}_{\mathrm{N+R-1}} \cdot \mathrm{DV}_{\mathrm{N+R-2}} \cdot \ldots \cdot \overline{\mathrm{DV}_{\mathrm{N+R-j}}}
\end{aligned}
\tag{2}
$$

To illustrate serial data transmission, let us consider a *CSL* with four data bits X_3,X_2,X_1 and X_0, four regular TSVs W_4, W_3,W_2,W_1 and one spare W_0, of which wires W_4 and W_1 are faulty. Therefore, there are only three functional TSVs and data transmission takes two cycles. In the first cycle, data bits X_3, X_2 and X_1 are sent on the first available functional TSVs starting from W_4, W_3, and W_2, respectively: W_3, W_2 and W_0. The upstream crossover switch control signals T_0 and flip-flop enable signals E_0 are represented in Figure IV-17 (a). In the second cycle, the remaining data bit X_0 is sent on wire W_0, as it is the first functional wire on which X_0 can be mapped. The control signals for this transmission cycle are represented in Figure IV-17 (b). The X values in Figure IV-17 indicate that the crossover switch has no corresponding switching element for that position and no control signals are needed.

$$T^0 = \begin{bmatrix} T_{30}=0 & T_{31}=1 & T_{32}=0 & T_{33}=0 & X \\ X & T_{20}=0 & T_{21}=1 & T_{22}=0 & T_{23}=0 \\ T_{13}=0 & X & T_{10}=0 & T_{11}=0 & T_{12}=0 \\ T_{02}=0 & T_{03}=0 & X & T_{00}=0 & T_{01}=0 \end{bmatrix} \quad T^1 = \begin{bmatrix} T_{30}=0 & T_{31}=0 & T_{32}=0 & T_{33}=0 & X \\ X & T_{20}=0 & T_{21}=0 & T_{22}=0 & T_{23}=0 \\ T_{13}=0 & X & T_{10}=0 & T_{11}=0 & T_{12}=0 \\ T_{02}=0 & T_{03}=0 & X & T_{00}=0 & T_{01}=1 \end{bmatrix}$$

$$E^0 = \begin{bmatrix} E_3=1 & E_2=1 & E_1=1 & E_0=0 \end{bmatrix} \quad E^1 = \begin{bmatrix} E_3=0 & E_2=0 & E_1=0 & E_0=1 \end{bmatrix}$$

(a) (b)

Figure IV-17 *CSL* control signals for the two-cycle transmission of 4 data bits on 3 functional TSVs: (a) the crossover switch and FF enable signals T_0 and E_0 for the first cycle and (b) the crossover switch and FF enable signals T_1 and E_1 for the second tranmission cycle

The CSL costs depend both on the TSV technology and the serialization / deserialization circuitry area. Using the grouping strategy, it is possible to reduce the complexity of the serialization circuitry and reduce CSL costs. The grouping-based cost reduction strategy is presented in the following.

4.2.2.3 Signal Grouping

In order to reduce *CSL* complexity, let us consider that the N data signals of each link are split into N_G groups. For each group of N_j regular TSVs, a number of R_j spares are allocated. The *CSL* functionality is now separated in a serializing part and a signal remapping part. In Figure IV-18, the partitioned upstream crossover switch of a *CSL* with N data bits and $N_G=4$ groups is represented.

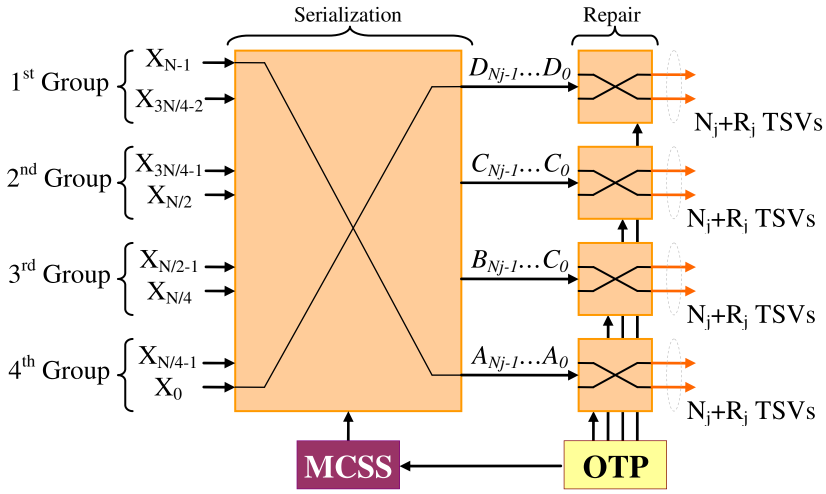


Figure IV-18 Logical decomposition of the upstream crossover switch for CSLs with 4 groups

Within each of the four groups *A*, *B*, *C*, and *D*, only spare-based repair is performed (i.e. no serialization). Groups are considered functional if the number of faulty TSVs does not exceed the number of spares. Hence, only functional groups are used to transmit the data in several transmission cycles. The repair stage in Figure IV-18 comprises several crossover switches for performing TSV repair in each group. Its control signals take constant values during all transmission cycles. For each of the data signal groups $\{A_{Nj-1},\ldots,A_0\}$, $\{B_{Nj-1},\ldots,B_0\}$, $\{C_{Nj-1},\ldots,C_0\}$ and $\{D_{Nj-1},\ldots,D_0\}$, the crossover switch control signals come directly from the OTP memory.

The serialization stage in Figure IV-18 comprises a crossover switch for connecting the data bits corresponding to the repaired and unrepaired groups to the positions corresponding to the repaired TSV groups. The control signals of this crossover switch change at each of the K cycle of a serial transmission. Thus, the *MCSS* selects the appropriate set of signals from *OTP* for each transmission cycle.

Similar to the optimization process of *TSV-SnR*, it is possible to find a grouped *CSL* configuration with minimal area. Given the number of regular TSVs n, the maximum number of spares r_{MAX} and the minimum number of functional groups g_{min}, the algorithm in Figure IV-19 determines the partitioning such that the *CSL* area is minimal and the target yield is achieved.

```
01: for g=1…g_min
02:     [n_1,…,n_g] ≈ ⌊n / g⌋;
03:     [r_1,…,r_g] = 0;
04:     Y = Y_TSV^n;
05:     while (Y < Y_T)
06:         i = ((i+1) mod g) + 1;
07:         r_i = r_i + 1;
08:         Y =Yield(g, g_min, n_1, r_1,…, n_g, r_g);
09:     end;
10:     area = estimate_area(n_1,…,n_g; r_1,…,r_g);
11:     if (area < smallest_area) then
12:         store(n_1,…,n_g; r_1,…,r_g);
13:     end;
14: end;
```

Figure IV-19 Area optimization process for CSLs

Starting with one group, the regular TSV are split in quasi-equal groups, since a uniform fault distribution is considered. In lines 05-09, spares are allocated for each group $n_1,\dots,n_g$ until the reliability target is achieved. The interconnect yield is estimated for each group, assuming that at least g_{min} groups are functional. For the uncorrelated fault model, a formula for the *Yield* function in line *08* can be found in Chapter 5 (i.e. Equations V-4, V-5). For each group configuration, the *CSL* configuration area is estimated and, if this area is smaller, it is stored. At the end of this process, the *CSL* configuration with the smallest area is returned.

4.2.3 Interconnect Built-In Self-Test, Self-Repair and Adaptive Serialization

A major limitation of the *TSV-SnR* and *CSL* is that only structural faults due to manufacturing defects can be repaired. In order to ensure high 3D NoC link TSV reparability for permanent faults due to interconnect aging and wear-out, a strategy for on-chip TSV test and repair is necessary. In this section, this issue is addressed by the built-in self-test and spare-/serialization-based repair (*IBIRAS*) strategy.

4.2.3.1 IBIRAS Architecture

The self-repair and adaptive serialization techniques are jointly used in *IBIRAS* to ensure inter-die communication in 3D integrated systems using highly defective vertical wires. Similar to *TSV-SnR*, interconnect built-in spare-based self-repair (i.e. *IBISnR*) consists in replacing faulty regular TSVs with fault-free spares. However, in the case of highly defective TSVs, this technique cannot efficiently maintain high reparability levels, as functional spares may run out. To this end, *IBIRAS* uses adaptive serialization to repair inter-die links that cannot be fixed using spares. The *Interconnect BIST*, *IBIRAS Reconfiguration Logic*, *Serialization*, and *Deserialization* modules for an *n*-bits inter-die link are represented in Figure IV-20.

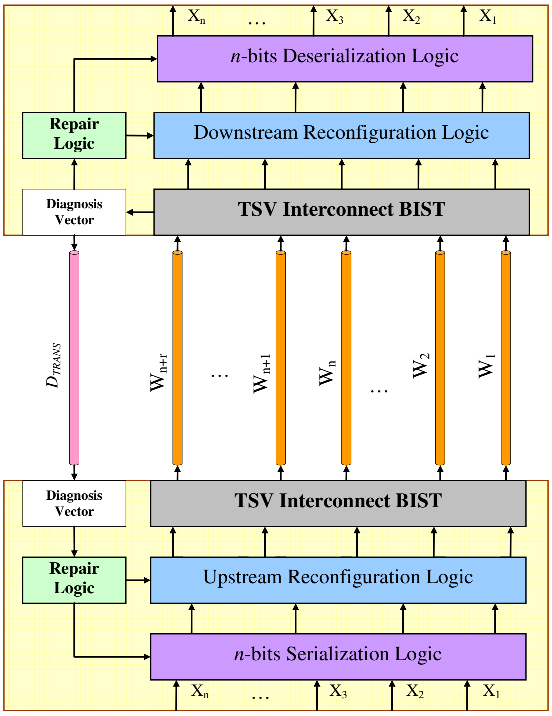


Figure IV-20 Self-repair and adaptive serial link

The self-repair and serialization/deserialization signals have to be generated on both sides of the inter-die link. Therefore, after TSV interconnect tests, the diagnosis vector *DV* must be stored on both sides of the link. As mentioned in Chapter 3, the diagnosis vector *DV1-DVn+r* is stored in the response analysis flip-flops FF_1-FF_{n+r} (upper die in Figure IV-20). From these *FF*s, the diagnosis vector is serially transmitted through D_{TRANS} into the lower-die register. This operation is done by configuring the upstream *DV* register as a feed-back shift register and the downstream register as a shift register. Thus, $n+r$ cycles after the interconnect tests, DV_1-DV_{n+r} is stored in both layers where it is further used to determine the *IBIRAS* repair signals.

The parameters of the design are the number of data bits n, the number of spare TSVs r, and the minimum acceptable number of fault-free TSVs m_{LIMIT} which determines the maximum acceptable number of serialization cycles K_{MAX}. If the number of fault-free TSVs is less than m_{LIMIT} then the link is assumed failed. The interconnect test diagnosis vector *DV* obtained in the test phase is used to determine the *IBIRAS* self-repair and adaptive serialization control signals. The upstream and downstream *Reconfiguration Logic* modules replace faulty regular TSVs by fault-free spares. If the number of fault-free TSVs m is less than n and more than m_{LIMIT} then the serialization / deserialization circuitry is enabled. Messages are serialized in the upstream interface by mapping only m out of n data bits on m fault-free TSVs until all the n bits of the message are transmitted. At the downstream interface, the received data bits are stored in a Deserialization Register in order to recreate the original message.

Even if an inter-die link is reparable (i.e. $m \geq mLIMIT$), the self-repair correctness may be compromised if errors occur during the diagnosis vector serial transmission from the upper to the lower layer. Faults on the *DTRANS* inter-die connection cause the repair logic to generate inconsistent repair signals. This connection is critical and protection against transient and permanent faults is ensured using *triple modular redundancy*

(*TMR*). The *IBIRAS* technique is independent on the link flow control protocol. The link control signals are routed on fault-free TSV in priority.

4.2.3.2 *Self-repair circuitry*

The self-repair strategy of *IBIRAS* consists in shifting each input signal b_i by one or more positions, until reaching the first fault-free TSVs that is not yet occupied by another signal. The Reconfiguration Logic consists in a crossover switch and a combinational logic block that generates the switch control signals. In Figure IV-21, the MUX-implementation of the crossover switch is illustrated for a 4-bits link with four regular TSVs, W_1-W_4, and one spare W_5. In this example, the link is functional if there are at least 2 fault-free TSVs (i.e. $m_{LIMIT} = 2$). In the worst case, flits are sent in two transmission cycles.

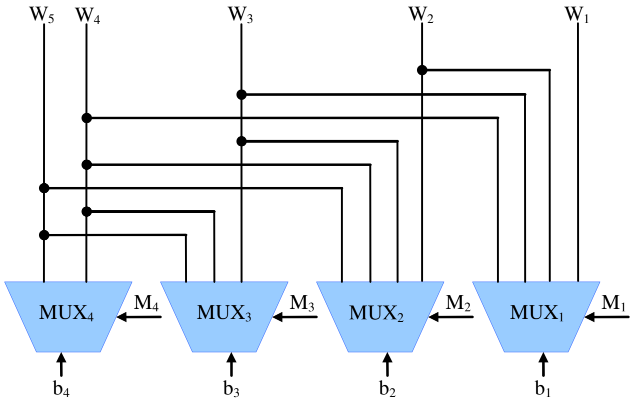


Figure IV-21 MUX implementation of the crossover switch in the reconfiguration logic

For n regular and r spare TSVs, the upstream *Reconfiguration Logic* crossover switch has n inputs and $n+r$ outputs. If all regular TSVs are fault-free, each bit b_i of the message is transmitted on TSV W_i. The crossover switch must be able to shift each input b_i by $n+r-m_{LIMIT}$ positions for $1 \leq i \leq m_{LIMIT}-1$, and by $n+r-i-1$ positions for $m_{LIMIT} \leq i \leq n-1$, as the link is considered functional if there are at least m_{LIMIT} fault-free wires. The MUX_i control signals M_i are encoded as *one-hot* codes and they represent the number of positions each input b_i must be shifted. Using the M_i^j notation (i.e. the j^{th} signal of MUX_i control signal M_i), for each input b_i with $0 \leq i \leq m_{LIMIT}-1$, the M_i MUX control signals are M_i^1-$M_i^{n+r-mLIMIT+1}$; for each b_j with $m_{LIMIT} \leq j \leq n$, the M_j MUX control signals are M_j^1-M_j^{n+r-j}. The downstream *Reconfiguration Logic* is mirrored, as data signals received on the $n+r$ TSVs are mapped on the downstream crossover switch n outputs.

The MUX control signals are computed using the diagnosis vector DV_1-DV_{n+r}. The equations of the control signals of MUX_1 are simple, as the repair process starts from signal $b1$. These equations are:

$$M_1^1 = \overline{DV_1}, M_1^2 = \overline{DV_2} \cdot DV_1, \ldots, M_1^{r+1} = \overline{DV_{r+1}} \cdot \overline{DV_r} \cdot \ldots \cdot DV_1 \qquad (3)$$

The equations for MUX_i control signals are more complex, as they have to take into account the fault-free TSVs already occupied by other positions. To reduce the cost of the logic generating them, iterative equations, which generate the control signals of position $i+1$ by using the control signals of lower order positions, are used. These equations are given below:

$$M_{i+1}^j = \overline{DV_{i+j+1}} \cdot \left(M_i^j + M_i^{j-1} \cdot DV_{i+j} + M_i^{j-2} \cdot DV_{i+j-1} \cdot DVi + j + \ldots + M_i^1 \cdot DV_{i+1} \cdot DV_{i+2} \cdot \ldots \cdot DV_{i+j}\right) \qquad (4)$$

The iterative equations (3,4) enable low-area hardware implementations of the repair function, but their recursive nature induces large delays. The *DV* values of their inputs are computed just once after the test and diagnosis phase and remain constant until the next test phase. They stay unchanged during circuit normal operation and their delays do not impact the link timing.

4.2.3.3 Adaptive serialization circuitry

Link flow control signals are repaired in priority and they are connected to the less significant positions of the self-repair MUXes. As the number of control signals is small compared to the total number of TSVs, self-repair will ensure their repair. However, it is possible that the number of fault-free TSVs is insufficient for repairing the data signals. In this case, the serialization circuitry is activated to transfer data signals in several cycles. In Figure IV-22, the serialization and de-serialization circuitry is represented.

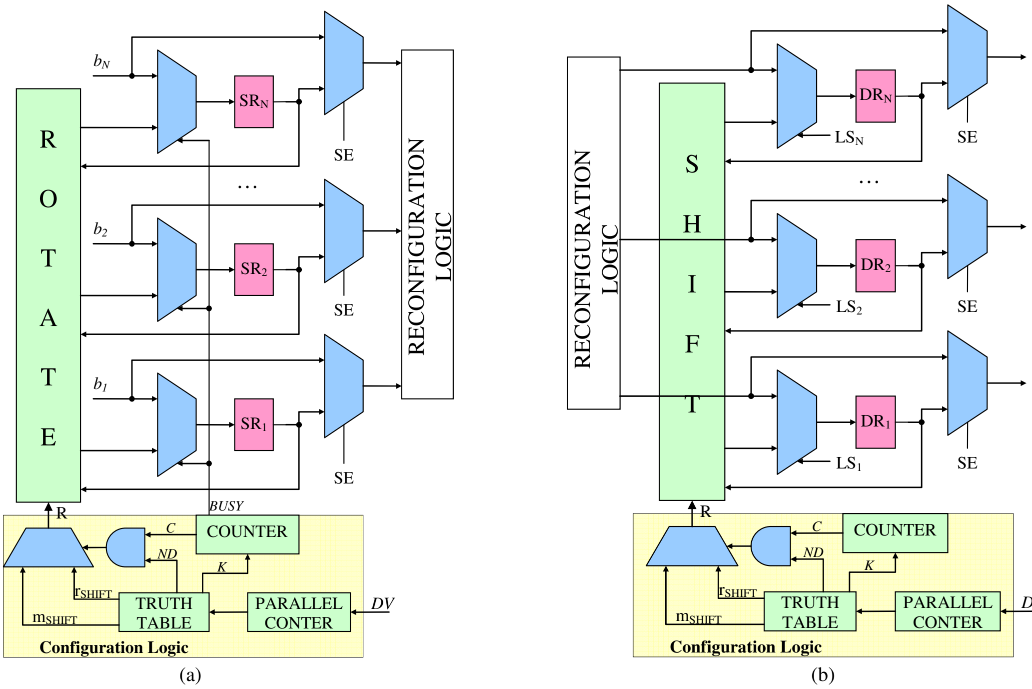


Figure IV-22 *n*-bit Serialization (a) and Deserialization (b) modules without extra rotate cycle

Each of the serialization/deserialization modules comprises a configuration logic block that determines the internal control signals like the interface statues (*BUSY*), number of positions to rotate / shift the serialization / deserialization register content. These signals are determined using the interconnect test diagnosis vector *DV*. The main phases of the adaptive serialization scheme are: load data bits in serialization register (*SR*), serial transmission from *SR* to deserialization register (*DR*), register contents rotate / shift and forward message from *DR* downstream.

In the first transmission cycle of the serialization/de-serialization process, the values coming from the m_{SHIFT} right-most positions of the serialization register *SR* are transmitted and loaded into the m_{SHIFT} left-most positions of the Deserialization Register *DR*. At the same time, *SR* rotates its contents by m_{SHIFT} positions to the right to prepare them for the next transmission cycle. In the second transmission cycle, the m_{SHIFT} right-most positions of *SR* are loaded into the m_{SHIFT} left-most positions of *DR*, which also shifts its contents by m_{SHIFT} positions to the right. At the same cycle *SR* rotates its contents by m_{SHIFT} positions to the right. This is repeated at each transmission cycle except for the last rotation of *SR* (i.e. the one performed at cycle $\lceil n/m \rceil$ - 1), which prepares the contents of SR for the last transmission cycle (cycle $\lceil n/m \rceil$). During this cycle *SR* rotates its contents to the right by R positions instead of m_{SHIFT} positions, with $R = m_{SHIFT}$ if m_{SHIFT} divides n and $R = r_{SHIFT} = n \bmod m_{SHIFT}$ otherwise. Loading the incoming bits in the m_{SHIFT} left-most positions of the deserialization register *DR* instead of the m_{SHIFT} right-most positions (used in the previous section) is equivalent to rotating its content by m_{SHIFT} positions to the right. Based on this observation, the extra rotation of m_{SHIFT} positions can be eliminated, and the message is transmitted in $\lceil n/m \rceil$ cycles, instead of $\lceil n/m \rceil+1$.

In order to load data from the m_{SHIFT} right-most positions of *SR* into the m_{SHIFT} left-most positions *DR*, the upstream Reconfiguration Logic is implemented to connect fault-free TSVs to its left-most outputs. As the reconfiguration is driven by *DV*, which points to the positions of the *m* fault-free TSVs, this solution will connect the *m*-fault free TSVs to the *m* left-most outputs of the upstream Reconfiguration Logic. To resolve this issue, a sequential circuit that modifies the content of the *DV* register by setting to *'1'* all positions of *DV* after the first right-most positions containing *mSHIFT '0's*. This way, the Reconfiguration Logic will consider all other positions as faulty and will connect the m_{SHIFT} right-most fault-free TSVs to its m_{SHIFT} left-most outputs.

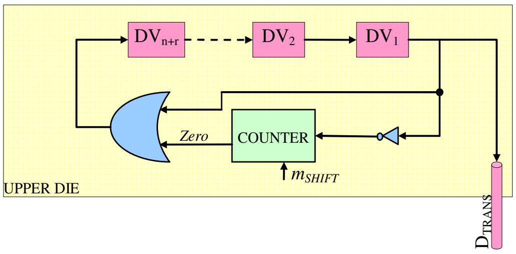


Figure IV-23 Circuit for diagnosis vector modification

In the circuit represented in Figure IV-23, the *n+r* flip-flops of the downstream *DV* Register are connected into a shift register configuration. The value m_{SHIFT} is loaded in the counter. The counter decrements each time signal DV_1 is *'0'*. The Signal *Zero* decodes the all *0*s state of the counter. When *Zero* becomes '*1'* it holds the counter, blocking it at its all *'0'*s state. After *n+r* shifts, the state of the diagnosis vector *DV* has been modified to *DV'*. The above operations are performed once after the test and diagnosis phase. Thus, computing *DV'* they will not delay system operation, as it can be performed during *DV* serial transmission between dies.

In this section, data link solutions for TSV permanent faults have been presented. These solutions are based on well-known spare-based and serialization strategies. While *TSV-SnR* and *CSL* are aimed at TSV faults due to manufacturing, the proposed *IBIRAS* approach can be used with an efficient *IBIST* scheme to ensure high TSV reparability during system lifetime.

4.3 *Network error resilience for transient faults*

In Section 4.1, the transient faults that affect flits traversing inter-die links are mitigated using data link error control schemes. Within 3D NoCs, flits traverse many routers, intra-die and inter-die links on their source to destination path. The network interface (*NI*) implements the transport and network levels, as it builds packets from transaction messages. These packets are further divided in flits that are sent through the interconnection network to destination. Thus, the network can be modeled as a *black-box* whose input/output ports are connected to the *NI*. Transients may affect flits anywhere along the path, reducing communication reliability. In this section, flit integrity is addressed by *network-level Forward Error Correction* (*NL-FEC*).

4.3.1 Network-level Forward Error Correction

In *NL-FEC*, flits are encoded at source using error correction codes (e.g. Hamming, Hsiao) before being sent on the network. At destination, transmission errors through the network are detected and corrected. Unlike link-level *FEC*, where only transients on the *PHY* are corrected, in *NL-FEC* transients on inter-die/intra-die links and routers are jointly mitigated. In Figure IV-24, a 3D Network protected using NL-FEC is represented.

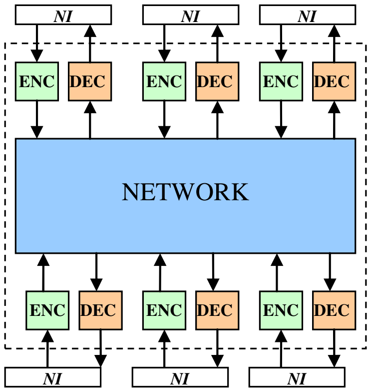


Figure IV-24 Network-Level Forward Error Correction

The encoding modules *ENC* are connected between the *NI* and the *network*. They append to each flit the error check bits of the error correction coding scheme. The decoding modules *DEC* check the flits leaving the network for errors. If errors are detected then they are corrected before the flit is forwarded to the *NI*.

While flit data bits are protected using error correction codes, special care must be taken about the side-band information (e.g. *End-of-Packet*, *Begin-of-Packet*). In *NL-FEC*, the side-band information bits are split in groups of up to two bits that are encoded using SEC codes. This choice for smaller group size is motivated by the fact that the side-band information is often used by routers during flit / packet processing. Hence, coding too many bits would affect router and network timing. If the *EoP* bit is encoded on three positions (i.e. *TMR*) then decoders (i.e. *majority voters*) are included in routers before the *EoP* is used. After the *EoP* flag is received, the router reallocates the output port to another input requesting that port. Hence, in the FSM controlling the output port, a majority voter is added for decoding the *EoP* field of the tail flit.

Major changes in terms of network circuitry and wiring are also due to the increased flit size. The size of router internal buffers increases, as both data bits and error control bits are stored. More intra-die wires and TSVs are necessary for parallel encoded flit transmission.

Transients on links and routers are corrected using Hamming SEC codes. If one clock cycle is necessary for data encoding, error detection and error correction, then the flit latency from source to destination increases by one clock cycles when no errors are detected and by two clock cycles when the detected error is corrected. Similarly to link-level FEC, the correction bypass mechanism is managed by a finite state machine (*FSM*). When an error is detected, the decoded switches off the bypass module in order to enable data correction and avoid data conflicts on the decoder outputs. After transmission ends, the decoder re-enables the correction bypass.

Because *NL-FEC* correction is performed only at the destination node, transients cumulate along the source-destination path, affecting network reliability. The network communication reliability is defined as probability that a flit traversing it arrives at destination fault free. The network reliability is defined as the minimum of the reliability of all paths in the network (i.e. 2-terminal reliability).

Given an *NL-FEC* protected network, the maximum flit size that can be reliably transmitted depends on the wire error rate ε_{wire}, router error rate ε_{router} and the targeted reliability level $R_T=1-\varepsilon_T$. When Hamming SEC codes are used, the probability of cumulating double errors along the path must be less than ε_T. In Figure IV-25, the flits size that can be reliable transmitted over a *10*-hops path for a router flit error rate of $\varepsilon_{router}=10^{-8}$ and uncorrelated faults.

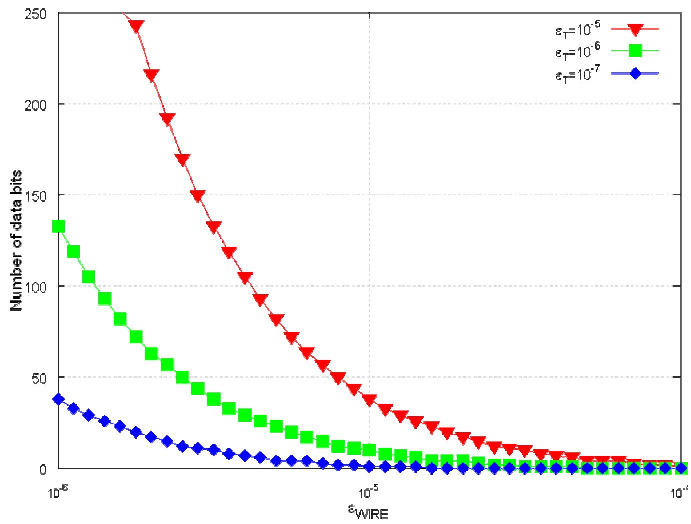


Figure IV-25 Number of data bits that can be reliably transmitted over a 10-hops path with a router error rate $\varepsilon_{router}=10^{-4}$

Increasing wire error rates and reliability targets make impossible the transmission of larger flits over long distances. Even for 10^{-6} wire error rates, it is not possible to send more than *80* data bits with less than one uncorrectable error per flit in 10^{7}. Therefore, powerful codes with multiple error correction may be necessary in order to ensure the target reliability levels. When transients on routers are negligible, fewer data bits can be sent with the same reliability targets through the *NL-FEC* protected network than in the case of data link *FEC*. For example, only 22 data bits can be sent using NL-FEC for a reliability target of 10^{-6} and a 10^{-5} wire error rate, while with FEC link protection up to 132 data bits can be reliably sent.

In order to cope with multiple errors that cumulate along the path, SEC codes are interleaved in order to increase the correction capabilities. However, the flit size, link width (i.e. wire count) and router overheads increase, as more error check bits are appended to each flit.

4.4 *Network error resilience for TSV permanent faults*

Permanent TSV faults do not affect only the inter-die link correct behavior, but the behavior of the entire network. Errors due to permanent faults could be managed at higher 3D NoC abstraction levels. Packets are routed from source to destination along designated paths. If the network contains faulty components (i.e. inter-die, intra-die links, and routers) then packets must be routed around these faulty components. The capability of avoiding faulty components is achieved by implementing *fault-tolerant routing algorithms*. In this section, a TSV fault-tolerant routing algorithm for 3D mesh NoCs is presented.

4.4.1 TSV-Fault Tolerant Routing in 3D NoCs

The proposed fault tolerant routing algorithm copes with TSV faults due to manufacturing defects. It assumes that *all* 3D NoC intra-die components (i.e. routers and intra-die links) are functional. When a router has a faulty inter-die port or when the adjacent inter-die link is faulty, packets must be rerouted on alternative fault-free paths. These alternative paths are not always minimal and they go through special nodes (i.e. *master nodes*) that have functional inter-die links in the requested direction. Depending on the link direction, these nodes are referred as *master-up* or *master-down* nodes.

To illustrate the routing algorithm, let us consider a regular 3D mesh topology with seven-port router. Four ports are used for intra-die communication (i.e. *NORTH*, *SOUTH*, *EAST*, and *WEST*) with neighboring routers, one port is used for communication with the local IP (i.e. *LOCAL*) and two ports for inter-die communication (i.e. *UP* and *DOWN*). In the fault-free case, each router has functional inter-die links and packets are routed using the ZYX routing algorithm. The TSV-fault tolerant routing algorithm requires an extra register per router to store the intra-die coordinates of its *master-up* (X_{UP}, Y_{UP}) and *master-down* (X_{DOWN}, Y_{DOWN}) nodes. In Figure IV-26, the routing algorithm for the node with coordinates (X_{LOCAL}, Y_{LOCAL}, Z_{LOCAL}) is given.

```
01: if (ZLOCAL = ZDEST) then
02:     if (YLOCAL = YDEST) then
03:         if (XLOCAL = XDEST) then OUTPUT(LOCAL);
04:         elsif (XLOCAL > XDEST) then OUTPUT(NORTH);
05:         else OUTPUT(SOUTH);
06:         end if;
07:     elsif (YLOCAL > YDEST) then OUTPUT(EAST);
08      else OUTPUT(WEST);
09:     end if;
10: elsif (ZLOCAL > ZDEST) then
11:    if(YLOCAL = YDOWN) then
12:       if(XLOCAL = XDOWN) then OUTPUT(DOWN);
13:       elsif (XLOCAL > XDOWN) then OUTPUT(NORTH);
14:       else OUTPUT(SOUTH);
15:       end if;
16:    elsif (YLOCAL > YDOWN) then OUTPUT(EAST);
17:    else OUTPUT(WEST);
18:    end if;
19: else
20:   if(YLOCAL = YUP) then
21:       if(XLOCAL = XUP) then OUTPUT(UP);
22:       elsif (XLOCAL > XUP) then OUTPUT(NORTH);
23:       else OUTPUT(SOUTH);
24:       end if;
25:    elsif (YLOCAL > YUP) then OUTPUT(EAST);
26:    else OUTPUT(WEST);
27:    end if;
28: end if;
```

Figure IV-26 TSV-fault-tolerant Routing algorithm in 3D meshes

If the packet is in the destination layer *($Z_{LOCAL} = Z_{DEST}$)* then, in lines 02-08, it is routed on the Y and X directions to the destination node *(X_{DEST}, Y_{DEST}, Z_{DEST})*. If the packet is not in the destination layer then it should go *UP* if $Z_{LOCAL}<Z_{DEST}$, or *DOWN* if $Z_{LOCAL} > Z_{DEST}$. If the routing direction is *UP*, but the local node does not have a functional *UP* link $(X_{UP} \neq X_{LOCAL}, Y_{UP} \neq Y_{LOCAL})$, then, in lines 11-18, the packet is routed on the Y then the X direction to its designated master node (X_{UP}, Y_{UP}).

If the routing direction is *DOWN*, but the local node does not have a functional *DOWN* link $(X_{DOWN} \neq X_{LOCAL}, Y_{DOWN} \neq Y_{LOCAL})$, in lines 21-27, the packet is routed on the Y then the X direction to its designated *master-down* node (X_{DOWN}, Y_{DOWN}). This process is reiterated by each node until the destination layer Z_{DEST} is reached. In Figure IV-27, the routing algorithm is illustrated for two packets, from *A* to *B* and from *C* to *D*, traversing the 3×4×3 NoC with functional inter-die links (a) and some faulty inter-die links (b).

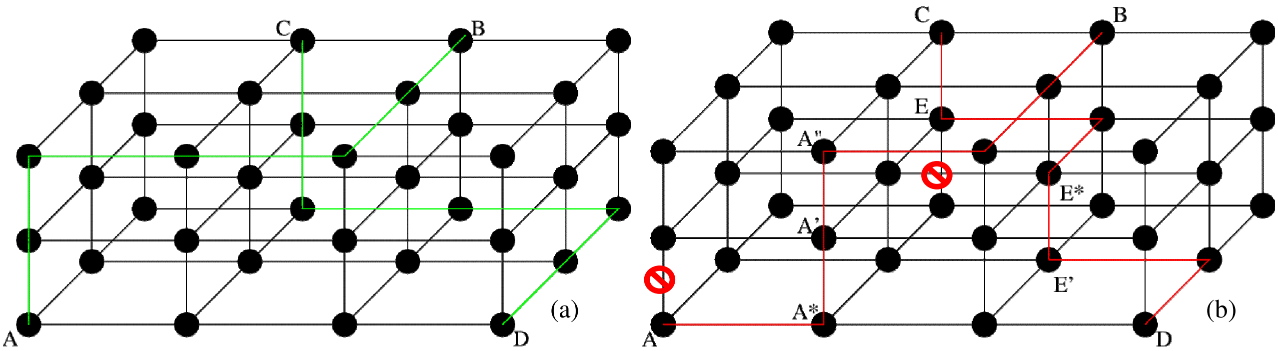


Figure IV-27 3x4x3 3D mesh topology

The *A-to-B* packet hops to *A**, the master-up of mode *A*, as its up link is faulty. Then it is transmitted to node *A'* in the second die, and then to node *A"* in the top destination layer. From *A"* it is routed to destination node *B* using the *YX* algorithm.

The *C-to-D* packet hops on the down vertical link to node *E*. However, it cannot continue on this direction, since the down link of node *E* is faulty. Therefore, the packet is directed to master-down node *E**. Here, it hops to node *E'* where it is routed to the destination node *D*.

It can be noticed that in both cases the fault-free paths have the same number of hops as the initial minimal paths. In other words, there is no performance penalty. However, this is not always the case. For example, the fault-free *C* to *A* path in Figure IV-27 (b) has two extra hops, as the master-down node of *E* is *E**.

The network cannot be considered functional just because there is a fault-free path between each pair of nodes. Deadlocks are cyclic dependencies between routed packets that are difficult to prevent and detect. In a deadlock situation, two or more packets wait for one another to free the buffers allocated to them. However, these packets cannot continue because they are in a cyclic dependency. They block the entire network and could lead to system failure.

When packets are deviated from their normal paths, in order to avoid faulty inter-die links, deadlocks could appear. In many cases, deadlock is prevented by separating the traffic traveling up and down the stack using virtual channels [RAA10]. This solution has non-negligible costs, as the buffering resources of routers almost double.

Instead of using virtual channels, it is possible to select master nodes such that there are no link cyclic dependencies [DT04]. Given a number of functional inter-die links, a master nodes allocation strategy is presented in following section.

4.4.2 *Master Node Selection*

The master node selection (*MNS*) process consists in choosing the *master-up* and *master-down* nodes for all routing nodes such that the network is fully-connected (i.e. all functional nodes are able to communicate), deadlock-free, and the performance penalty is minimal.

Deadlock freedom is a critical property of NoCs ensuring that packets traversing it are not blocked due to a cyclic link (or channel) dependency. For packet-switched wormhole networks, deadlock freedom can be proven using the Link Dependency Graph (*LDG*). The *LDG* is an oriented graph attached to a network that describes the inter-link dependencies for all network paths. The way to guarantee deadlock-freedom is to prove that there are no cycles in the network's *LDG*.

The proposed TSV-fault-tolerant routing algorithm is not deadlock-free for any master node selections. In Figure IV-28 (a), a network configuration that satisfies the connectivity requirements (i.e. there is a path between every nodes) is presented. The 2×2×2 mesh has only two functional inter-die links (i.e. from N_2 to N_6 and from N_7 to N_3), and it uses the modified ZYX routing algorithm with packets travelling toward the destination layer first and then to the destination node using YX routing. This 3D NoC is not deadlock-free, as the *LDG* is not cycle-free. Let L_{ij} be the link connecting nodes N_i and N_j. In Figure IV-28 (b), the L_{12}-L_{26}-L_{68}-L_{87}-L_{73}-L_{31}-L_{12} cycle in the NoC's *LDG* is represented.

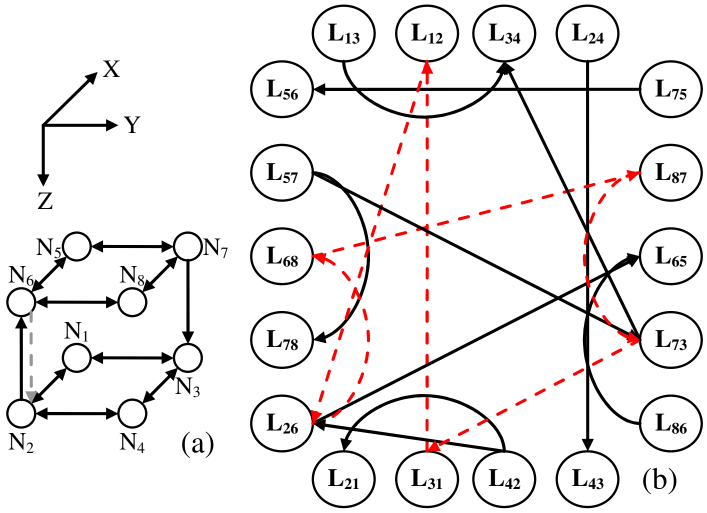


Figure IV-28 3D mesh configurations having cycles in its LDG

If two packets are simultaneously transmitted from nodes N_6 to N_2 (i.e. N_6-N_8-N_7-N_3-N_1-N_2) and nodes N_3 to N_7 (i.e. N_3-N_1-N_2-N_6-N_8-N_7) then these packets will create a deadlock, blocking communication between other nodes.

The solution to the deadlock problem is to find alternative master nodes such that the *LDG* is cycles-free. Using the *backtracking* strategy, alternative master nodes that lead to a deadlock-free configuration could be found. The master selection process minimizes performance penalty (i.e. network latency) by taking into account the distance to master nodes and inter-die link latency. Network latency is quantified using the zero-load-latency (*ZLL*) model. In *ZLL*, the packet latency is determined by the delays of routers and links along the source-destination path, without taking into account packet contention. In Figure IV-29, the selection process algorithm is summarized for *N* routing nodes.

```
01: i = 1;
02: while(i > 0)
03:     allocate_node(i);
04:         if allocation_exists then
05:             if i = N then
06:                 if deadlock_free_configuration() then
07:                     if performant_configuration() then
08:                         store_configuration();
09:                     end if;
10:                 end if;
11:             else
12:                 i++;
13:             end if;
14:         else
15:             i--;
16:         end if;
17: end while;
```

Figure IV-29 Master Node Selection Algorithm

In the algorithm, the master node selection process starts from node 1. The master-up/down nodes of the current node i are allocated in line 03. If these masters exist and master nodes have been allocated for all nodes (i.e. $i=N$) then the master node allocation is checked for deadlocks. If this configuration is deadlock-free (*deadlock_free_configuration()* returns *true*) and it has better average latency than previous configurations (*performant_configuration()* returns *true*), then it is kept as the optimal configuration (see lines 06-10). If there are unallocated nodes (line 11) then the sequence in lines 03-16 is reiterated for the next node $i+1$. If no allocation exists for the current node i then the algorithm backtracks to the previous node i-1, reallocate a new set of master nodes and reiterate lines 02-16.

Regular nodes (i.e. nodes without functional inter-die links) can choose any master node within the same layer. In the *allocate_node()* procedure in Figure IV-29, each regular node goes through all its intra-die master nodes. A solution always exists, as long as there is a pair of routing nodes with both inter-die links functional. If a solution exists it will be found, but the execution times could be very long. For the NoC in Figure IV-28 (a), no solution is possible, unless a functional link between N_6 and N_2, or N_3 and N_7 exists.

In order to reduce the execution time, a possible solution is to stop at the first deadlock-free configuration found. However, there is no guarantee that network average latency is minimal. In order to minimize latency, in the master selection process priority is given to the closest nodes with the fastest inter-die links. Hence, the potential master nodes M of each regular node N_L, are ordered using two parameters: the latency of the local node N_L to candidate master node N_i path and the inter-die link latency K_i. For the modified *MNS* algorithm, the solution consists of closest master nodes with the fastest links and a deadlock-free configuration. In Figure IV-30, the modified master node selection algorithm is presented.

```
i = 1;
while(i > 0)
    allocate_node(i);
        if allocation_exists then
            if i = N then
                if deadlock_free_configuration() then
                    store_configuration();
                    break;
                end if;
            else
                i++;
            end if;
        else
            i--;
        end if;
end while;
```

Figure IV-30 Fast Master Node Selection Algorithm

It is difficult to show that the *MNS* algorithms presented in this section are efficient, since there are no analytical formulas that predict their execution time. In order to have some time determinisms, it is possible to determine solutions for particular inter-die link fault patterns and store these solutions in a repository. After inter-die link tests, the status of each link is known. The NoC inter-die link fault distribution is then compared to that of the pre-determined solutions. If a best-fit solution is found then it is used for router configuration, otherwise the network is considered irreparable and the entire chip is discarded.

4.5 *Multi-layer Error Resilience for 3D NoCs*

In the previous sections a series of error resilience techniques for transient and permanent faults on TSVs have been presented. Each of these solutions is specific to an abstraction layer (i.e. data link and network) and NoC component (i.e. links and routers). When the system complexity and failure density increases, it becomes increasingly difficult to ensure correct system functionality with minimal costs (e.g. area and power overheads, TSV count) and without affecting system performance. This issue can be addressed using a multi-layer error resilience framework. In this section, the multi-layer error-resilience solutions to TSV permanent faults due to manufacturing defects and transits are presented.

4.5.1 Multi-layer TSV yield improvement

In the KGD stacking strategy, TSV manufacturing defects remain the main cause of 3D chip yield loss. In NoC-based 3D MPSoCs, link-level *spare-and-replace* and *configurable serialization* ensures high TSV reparability even when the amount of spares is very limited. TSV manufacturing defects can also be handled by repairing the network such that no source-destination paths go through inter-die link containing faulty TSVs. This network configuration process is performed off-chip, using the TSV test diagnosis vector.

In the multi-layer approach, data link (i.e. *TSV-SnR, CSLs*) and network-level solutions are jointly used in order to ensure high yield and reduce repair costs (i.e. number of spares). After final tests, 3D chips with faulty / un-reparable intra-die components (e.g. links, routers, network interface, or IP blocks) are discarded. If no such faults are detected then the TSV interconnect test diagnosis vector of all inter-die links is analyzed. In Figure IV-31, the diagram of the 3D NoC repair and configuration process is represented.

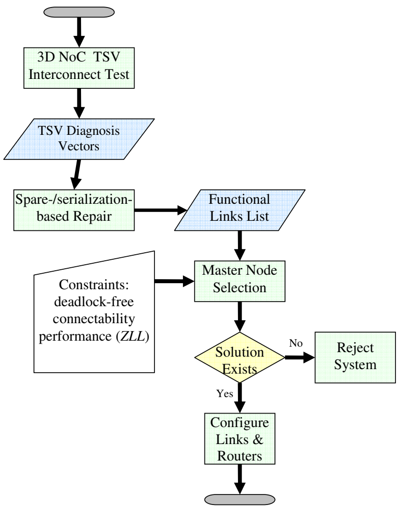


Figure IV-31 NoC-based 3D MPSoC configuration process

After the off-chip 3D NoC link TSV test and diagnosis, the link repair signals are computed. If no solution is possible (i.e. more faults that functional spares or no serialization is possible) then that link is diagnosed as faulty. After all links have been evaluated, the list of functional links with their serialization rate *K* is passed to the master node selection module together with different constraints (e.g. deadlock-freedom, fully-connected network, minimal latency). If a solution to the *MNS* problem exists then links and routers are configured by storing the link repair signals and master nodes coordinates in local *one-time-programmable* (*OTP*) memories.

This multi-layer scheme can repair many permanent faults due to manufacturing, as a solution to the *MNS* problem exists if there is at least one pair of routers between the stacked layers that are connected by two unidirectional links. Since inter-die links can now be configured to serialize data, the master-node selection process becomes more complex, as the serialization rate of each link must be taken into account. Ideally, the master of each regular node should be the closest node with the lowest serialization rate for the adjacent inter-die link.

4.5.2 Multi-layer reliability improvement

In the multi-layer reliability improvement strategy, data link-level and network-level error correction are jointly used for correcting transient faults during network (i.e. link and router) traversal. However, instead of re-encoding the already encoded flit, the error check bits computed at the source node are used for link-level error detection / correction. This way, the overheads are smaller, since no link-level encoding are necessary.

Of the three classes of link-level correction, only the *Forward Error Correction (FEC)* can be jointly used with *Network-Level-FEC* (*NL-FEC*). In the case of *ARQ*, if faults occur before the flit traverses the current link then a retransmission request is performed. Although Hamming codes have good error detection capabilities, it is not possible to recover by simple retransmission from such errors. In this case, retransmissions are made until the retransmission mechanisms times-out. Then, as data link error recovery is not possible, high-level recovery mechanisms must be used (e.g. packet dropping, transaction dropping and transaction retry). The *HYB* scheme will make a retransmission request only if the error cannot be corrected. In this case, *SECDED* codes must be used at network-level in order to avoid miss-correction. Like in the *ARQ* case, retransmission is effective only for faults occurring on the current link. If errors occur before then the retransmission mechanisms will time-out and high-level error recovery techniques must be used. Hence, these solutions may be suitable if and only if it is known that faults are more likely to occur on these protected links.

The joint data link-/network-level FEC scheme consists in adding a series of error detection and correction stages at some input ports of 3D NoC routers. In terms of reliability, this scheme is better than *NL-FEC*, as fault can cumulate only between two consecutive correction stages. The challenge of NL-FEC with link correction stages is how to select which links to protect. For 3D NoCs, correction could be implemented for inter-die links, as show in Figure IV-32.

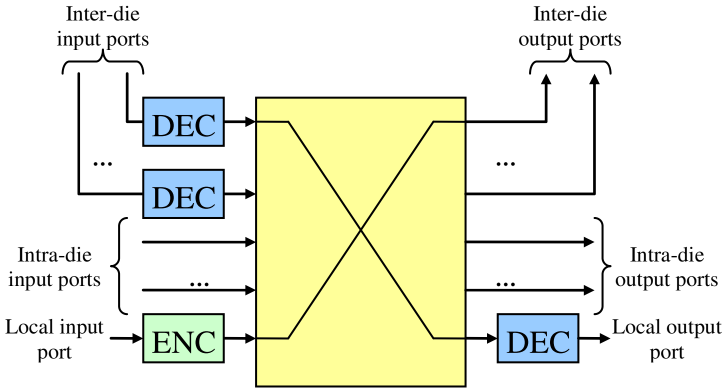


Figure IV-32 Network-level FEC protection with inter-die error detection/correction

Using this approach, errors can cumulate only within each layer, without contaminating neighboring layers. In terms of reliability, the error resilience of *NL-FEC* with inter-die correction stages depends on the intra-die communication reliability.

If intra-die reliability is not high enough then intra-die correction stages must be included. Without loss of generality, let us consider a 2D mesh XY intra-die network and the maximal path length p_{MAX} on which reliable communication can be achieved. Assuming Y-first routing, the maximal path length is $Y+X$ and there are four such paths. In order to ensure that there are no paths longer than p_{MAX}, a correction stage is added after every $p_{MAX}{}^{th}$ hop on every path in the network. In Figure IV-33, the intra-die correction stages of a 4×4 mesh with $p_{MAX}=4$ are represented.

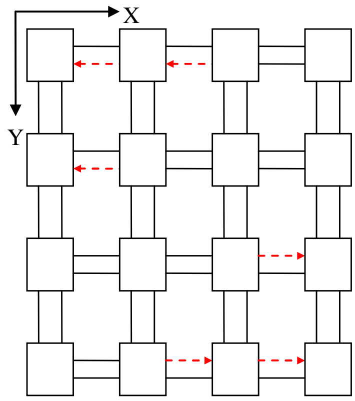


Figure IV-33 4×4 mesh NoC with correction stages every p_{MAX}=4 hop

The correction stage insertion algorithm analyzes every path in the network and adds a correction stage after the 4th hop from the last correction stage. Hence, a total of six correction stages are necessary within each 4×4 layer of the 3D NoC.

4.6 *Conclusion*

In this chapter a series of single- and multi-layer strategies for mitigating TSV permanent and transient faults have been presented. Permanent TSV failures due to manufacturing and aging/wear-out defects can be repaired using the spare-based approach (*TSV-SnR*) or using less conventional solutions such as serialization (*CSL*, *IBIRAS*) and fault tolerant routing (*TSV-FTR*). Transient faults can be detected and corrected using signal encoding combined with link-level retransmission (*ARQ, HYB*) or correction at data link (*FEC*) and network levels (*NL-FEC*). These techniques are only part of the solution. Individually, they could solve all the

TSV reliability and yield problems, but the costs and impact on network / system performance could be non-negligible for unreliable technologies. To this end, a multi-layer TSV error resilience strategy that uses these solutions is proposed.

In the following chapter, each error resilience strategy is fully assessed for a fully-synchronous 3D mesh NoC with different reliability / yield targets and interconnect failure rates. This analysis is aimed at identifying different fault-tolerance capabilities, costs and network performance trade-offs.

Chapter Five

EXPERIMENTAL EVALUATIONS OF ERROR RESILIENCE STRATEGIES

In 3D Networks-on-Chips, error resilience techniques are implemented at different abstraction levels. Each single/multi-layer strategy has its advantages and limitations with respect to the hardware costs (i.e. area / power overheads, number of redundant TSVs) and the impact on network performance and connectivity. In this chapter, the configuration process is presented for each error resilience scheme. The costs in terms of area / power overheads, TSV count and performance penalties are evaluated for different setups in order to identify trade-offs in single-/multi-layer strategies. The experimental assessments of the error resilience schemes are performed on seven-port routers used in 3D mesh NoC topologies.

5.1 *TSV Permanent Faults in 3D NoCs*

Manufacturing yield is one of the major roadblocks in getting TSV-based 3D integration mainstream. When known-good-die (*KGD*) bonding strategies are used, the inter-die connections (i.e. TSVs) are major contributors to the yield loss. In 3D NoCs, the TSV manufacturing yield of inter-die links is improved using spare-based (*TSV-SnR*) and serialization-based (*CSL*) repair. TSV faults can also be repaired at network level by configuring routers to send packets around faulty inter-die links (*TSV Fault-Tolerant Routing*). In this section, the repair capabilities, area overheads and impact on network latency of these error resilience schemes are evaluated.

5.1.1 ***Data link yield improvement strategies***

TSV manufacturing yield can be improved by allocating enough spare TSVs such that all faulty regular TSVs are replaced by functional spares (*TSV-SnR*). For highly defective technologies, allocating too many spares raises other challenges such as manufacturing costs due to higher number of TSVs / chip. In 3D NoCs, link-level serialization (*CSL*) is proposed as an alternative to spare-based repair. In the following, the repair capabilities of spares-based and serialization-based schemes are assessed. Different trade-offs for link-level repair with respect to the number of TSVs / chip and repair fabric area are also discussed.

5.1.1.1 TSV Spare-and-Repair

3D NoCs comprise tens or hundreds of inter-die links whose cumulated TSV faults may result in poor yield. TSV Spare-and-Replace (*TSV-SnR*) must ensure link reparability such that the link yield Y_{LINK} is greater than a yield target Y_T. Typical values for Y_T are well above *99%*, as the compound yield of *all* inter-die links must be high (i.e. > *98%*). Using the uncorrelated fault model, the yield of an n-bits link with r spares that can replace any faulty regular TSV can be expressed as [KM07]:

$$Y_{LINK} = \sum_{j=n}^{n+r} \binom{n+r}{j} \cdot Y_{TSV}{}^{j} \cdot \left(1 - Y_{TSV}\right)^{n+r-j} \qquad \text{(V-1)}$$

Using equation (V-1), the number of spares r per link is determined such that $Y_{LINK} \geq Y_T$. In other words, the fault tolerance capabilities of *TSV-SnR* protected links depend exclusively on the number of spares r. Let us consider that the cumulated link yield of a *20*-inter-die links 3D NoC must be at least 99%. Thus, the target yield of each link is Y_T=*99.95%* (i.e. 0.9995^{20} *~0.99*). From equation (V-1) it is determined that TSV failure rates up to d_{TSV}=*1% (i.e. YTSV=0.99*) can be handled with *r=3* spares/link for *n=32*-bits links, and *r=5* spares/link for *n=64*-bits links.

The main question one faces when using spare-based TSV repair is how to allocate spare resources for a TSV failure rate d_{TSV} such that the yield target is achieved and different constraints (e.g. repair fabric area or TSV count) are met. In the remaining of the section, the main focus is the optimal use of spare resources. To this end, the impact of the TSV technology on the number of spares necessary to achieve a target yield is assessed.

Let us consider an MPSoC with inter-die links having TSVs with *15µm* pitch. In order to assess the impact of TSV technology, the TSV pitch p_{TSV} is reduced from *15µm* to *10µm*, such that twice as many inter-die wires can be integrated without increasing the total TSV footprint. If the extra TSVs are used for increasing the throughput of each inter-die link (i.e. *double link data size*) then the number of links does not changes and each link must achieve its *99.95%* yield target, as the overall *20* inter-die links target yield is 99% (i.e. 0.9995^{20} *~0.99*). If the TSVs are used for increasing the number of links then the *link yield target increases*. In this case, there are twice as many inter-die links and each link must achieve a yield target of *99.975%* (i.e. 0.99975^{40} *~0.99*). In Figure V-1, the impact of the TSV technology on the number of spares, which is determined using equation V-1, is represented for 32-bits and 64-bits links.

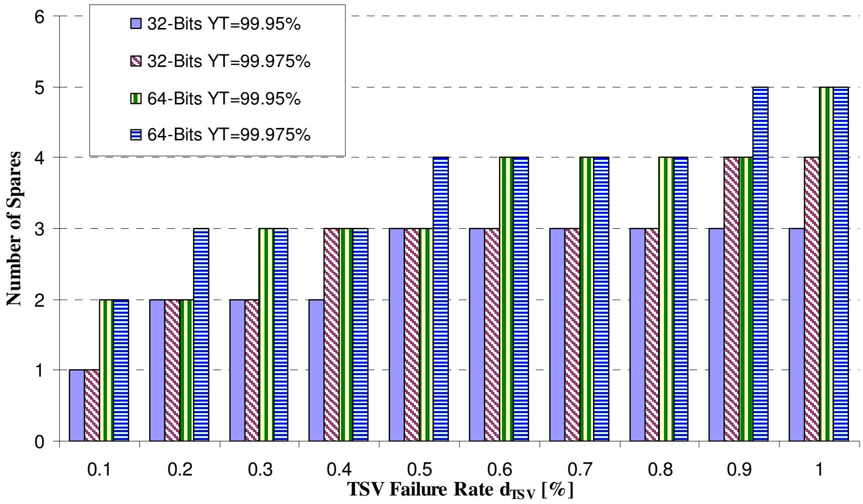


Figure V-1 Number of spares for 32-bits and 64-bits inter-die links

The number of spares *r* increases with the number of regular TSVs, the TSV failure rate d_{TSV} and the target yield Y_T. In the case of d_{TSV}*=0.1%*, the *99.95%* target yield is achieved using *r=1* spares for *n=32*, and *r=2* spares for *n=64*. The number of spares increases to *r=4* for 32-bits links and *r=5* for 64-bits links, when the failure rate is *1%* and the target yield is *99.975%* (i.e. 10 µm pitch and twice as many links / chip).

When the TSV density increases and the link width increases from 32 bits to 64 bits, the number of spares per link increases for the same target yield of 99.95% and failure rate d_{TSV} (see *32-Bits* Y_T*=99.95%* and *64-bits* Y_T*=99.95%* configurations in Figure V-1). In the case of d_{TSV}*=0.3%*, a total of 680 TSVs with 15µm pitch are used for 32-bit inter-die links. When the TSV density doubles, the number of 10 µm pitch TSVs necessary for 64-bit link with the same target yield increases to 1340 (i.e. less than double).

When the number of links increases, more spare TSVs could be necessary to achieve the higher *99.975%* target yield for the same failure rate d_{TSV} (see *32-Bits* Y_T*=99.95%* vs. 32-*bits* Y_T*=99.975, 64-Bits* Y_T*=99.95%* vs. *64-bits* Y_T*=99.975%*). For example, in the case of d_{TSV}*=0.3%*, the same number of spares is necessary to achieve the target yield (i.e. 2 spares for 32 bits links, and 3 spares 64 bits links). However, in the d_{TSV}*=0.9%* case, an extra spare is necessary for both 32 and 64 bits links. Moreover, the number of TSVs necessary to achieve the target yield could exceed that of available TSVs. In the case of 32 bits links, *700* TSVs with *15 µm* pitch are necessary to achieve the 99.95% target, while *1440* TSVs (i.e. more than double) are necessary for *10 µm* pitch TSVs.

There are few studies that show correlations between the TSV density and the TSV failure rate. Depending on how mature the TSV technology is, increasing the TSV density could also affect the TSV failure rate d_{TSV}. Ideally, integrating more TSVs has a negligible impact on d_{TSV}. In order to assess the impact of d_{TSV}, let us pessimistically consider that the TSV density and failure rates are proportional. Hence, doubling the number of TSVs per chip would double d_{TSV}. The *32*-bits inter-die link with d_{TSV}*=0.3%* needs only two spares to satisfy the target yield for the 3D NoC with *20* inter-die links (i.e. 680 *15µm* pitch TSVs). When the TSV density doubles, the TSV failure rate is d_{TSV}*=0.6%* and 64-bits links require *r=4* spares for the same 3D NoC configuration with *20* links (i.e. 1260 *10μm* pitch TSVs). If the number of links increases then the *99.975%* target yield is achieved by allocating $r=3$ spares per link (i.e. 1400 10μm pitch TSVs).

From these results it can be concluded that, in order to minimize the number of spares, the *TSV-SnR* strategy should be implemented for larger groups of regular TSVs (i.e. 3D NoC inter-die links should not be partitioned). Although allocating spares for more TSVs can reduce the TSV count, the *TSV-SnR* crossover switch size increases. This is due to the fact that, when there are fewer spares / chip, there are also more replace possibilities for each regular TSV.

In order to reduce the crossover switch complexity and minimize repair fabric area, the grouping strategy was proposed in the previous chapter. Let us consider that the n regular TSVs and r spares of each inter-die link are partitioned in g groups with $n_1 \ldots n_g$ regular TSVs (i.e. $n_1+\ldots n_g=n$) and $r_1 \ldots r_g$ spares (i.e. $r_1+\ldots+r_g=r$). The TSV yield is given by the probability that there are up to r_i faulty TSVs in each group i. Similarly to the previous case, the link yield using the uncorrelated fault model is expressed as:

$$Y_{LINK} = \prod_{i=1}^{g}\left(\sum_{j=n_i}^{n_i+r_i}\binom{n_i+r_i}{j}\cdot Y_{TSV}^{\ j}\cdot\left(1-Y_{TSV}\right)^{n_i+r_i-j}\right) \qquad \text{(V-2)}$$

In equation (V-2), when there are g groups and faults between groups are not correlated, the number of spares r_i is determined such the n_i regular TSV group yield is above $Y_{LINK}^{1/g}$. Let us consider a 32-bits link with a target yield $Y_T=99.95\%$. In Figure V-2, the number of spares for 32-bits links is represented for $g=1$, $g=2$, and $g=4$ groups, assuming single TSV failure rates d_{TSV} up to 1%.

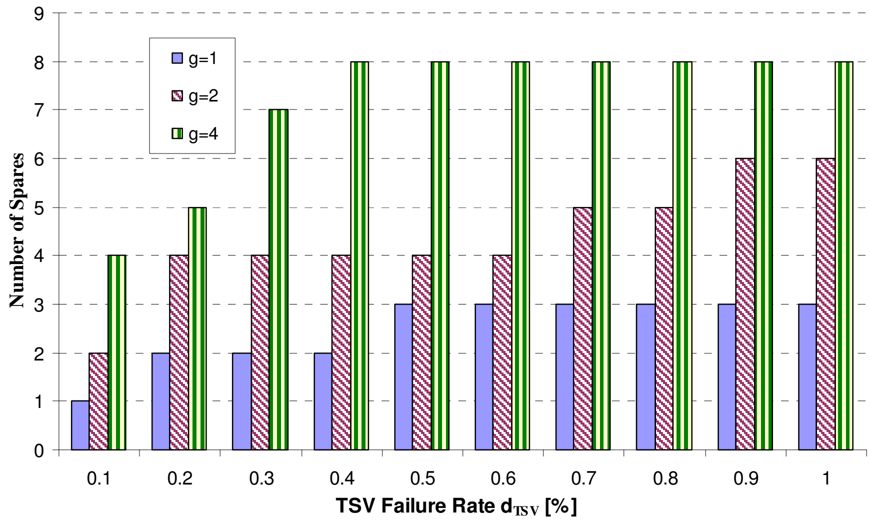


Figure V-2 Number of spares of 32-bits links when different grouping strategies are considered

Increasing the number of groups results in higher group yield targets that can only be achieved using more spares. When $d_{TSV}=1\%$, the number of spares increases from *3* to *8*, as there are $r_i=2$ spares for each group. In the $g=4$ case, each input signal can be mapped on three different wires: the initial regular TSV and two spares. Hence, the crossover switch complexity is reduced compared to the $g=1$ case, where each signal can be mapped on four different wires: the initial regular TSV and three spares. For the 3D NoC seven-port router, an assessment of area overheads is presented in section 5.1.1.3.

In 3D NoC links, *Spare-and-Replace* strategies have the advantage that, as long as signal propagation through the repair fabric and *PHY* is less than a clock cycle, they have no effect on network performance. In the hardware implementations, crossover switches are often implemented as arrays of tri-state buffers. Hence, the extra delays due to the repair fabric are often negligible.

The evaluations above indicate a major limitation of spare-based repair: it cannot always achieve the target yield with a limited number of spares. In 3D NoC inter-die links, these limitations are alleviated using repair strategies based on serialization.

5.1.1.2 Configurable Serial Links

The faulty TSVs of 3D NoC inter-die links can be repaired by implementing configurable fault-tolerant serialization (*CSL*) strategies. In this section, the fault tolerance capabilities of CSLs are assessed with respect to the TSV failure rate. Let us consider inter-die links with *N* data bits such that each TSV bundle comprises *N* regular TSVs and very few spares. In the *CSL* strategy, links are functional if at least M_{MIN} TSVs are functional. Hence, up to $F=R+N-M_{MIN}$ faults can be repaired. For uncorrelated TSV faults, the link yield Y_{LINK} is expressed as:

$$Y_{LINK} = \sum_{j=M_{MIN}}^{N+R} \binom{N+R}{j} \cdot Y_{TSV}{}^{j} \cdot \left(1 - Y_{TSV}\right)^{N+R-j} \qquad \text{(V-3)}$$

Equation (V-3) represents the probability that at least M_{MIN} out of $N+R$ TSVs are functional. Given the number of spares R, the number of minimal functional wires M_{MIN} is determined such that the link yield is above a target Y_T. Increasing the number of TSVs enables 3D NoC configurations with wider inter-die links or with more inter-die links. Unlike *TSV-SnR*, where the number of spares depends on the number of regular TSVs and the failure rate $d_{TSV}=1-Y_{TSV}$, in *CSLs* it is only M_{MIN} that changes, as the *N* and *R* values are design constants. In terms of costs, only the repair fabric complexity is affected, while the link throughput is reduced due to serialization in the case when there are more than *R* faults.

Let us consider 32-bits and 64-bits 3D NoC links with up to $R=4$ spares per link. In Figure V-3, the TSV yield of inter-die *CSLs* is represented for different configurations (i.e. number of spares *R* and minimum number of functional TSVs M_{MIN}).

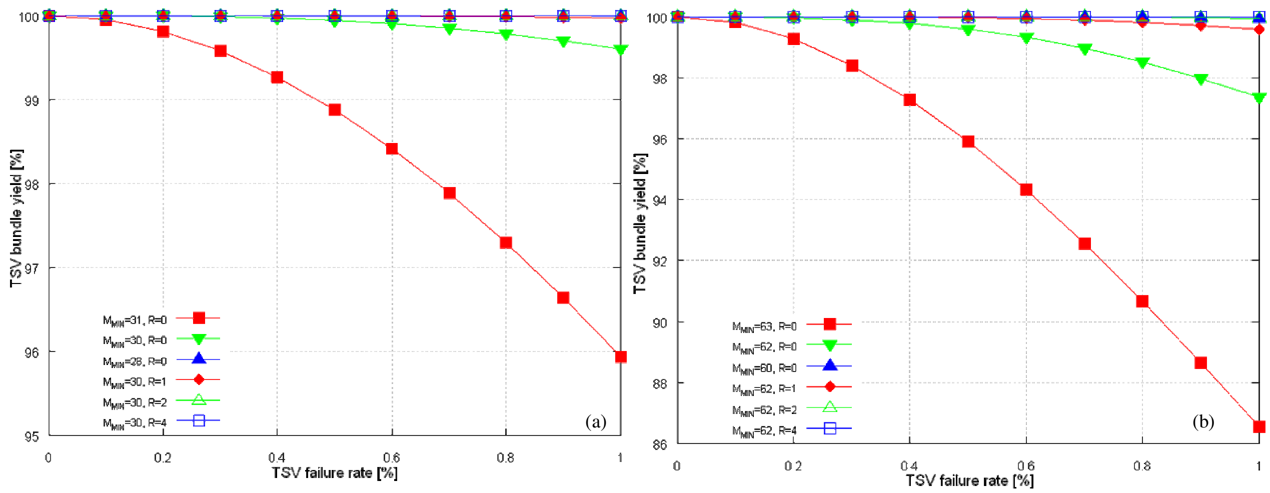


Figure V-3 TSV yield for 32-bits (a) and 64-bits (b) links using CSL

The results above show that the interconnect yield improves when the number of reparable faults F increases either by considering more spares or by reducing M_{MIN}. For a given F, the TSV yield is higher when there are fewer spares, as the probability of having more faults depends on the total number of TSVs per link (or bundle). Wider links (i.e. more regular and spare TSVs) have higher failure probability, as having more TSVs increases the probability than some of them are faulty. For example, being able to repair $F=2$ faults results in a *99.6%* TSV yield for 32-bits links, and *97.5%* for 64-bits links. Hence, for wider links to achieve similar yield levels, the repair capabilities (i.e. more spares or lower M_{MIN}) must be increased.

*CSL*s can achieve high yield targets without using spares even when $d_{TSV}=1\%$. However, the crossover switch complexity increases, since the minimal number of functional TSVs is reduced and the mapping capabilities for each wire increase.

Signal grouping was introduced as a way to reduce *CSL* complexity. Let us consider that the $N+R$ TSVs of a link are split in g groups of $n_1 \dots n_g$ regular TSVs and $r_{1 \dots} r_g$ spares such that the link is functional if at least g_{MIN} groups are functional. The TSV yield of group (n_i, r_i) is the probability that at least n_i TSVs are functional. For uncorrelated faults, the group yield Y_{group} is expressed as:

$$Y^i_{group} = \sum_{j=n_i}^{n_i+r_i} \binom{n_i+r_i}{j} \cdot Y_{TSV}{}^{j} \cdot (1-Y_{TSV})^{n_i+r_i-j} \qquad \text{(V-4)}$$

Without loss of generality, let us assume that these groups have the same yield: $Y_{group}{}^{1} = \dots Y_{group}{}^{g} = Y_{group}$. The *CSL*-protected link yield is the probability of having at least g_{MIN} functional groups. For uncorrelated TSV faults, it is expressed as:

$$Y_{LINK} = \sum_{i=g_{MIN}}^{g} \binom{g}{j} \cdot Y_{group}{}^{j} \cdot (1-Y_{group})^{g-j} \qquad \text{(V-5)}$$

In order to assess the impact of grouping on the *CSL* yield, let us consider 32-bits and 64-bits links with regular TSVs split in $g=2$ and $g=4$ groups, with spares distributed between these groups. In Figure V-4, the TSV yield is represented for different grouped *CSL* configurations having $g_{MIN}=1$.

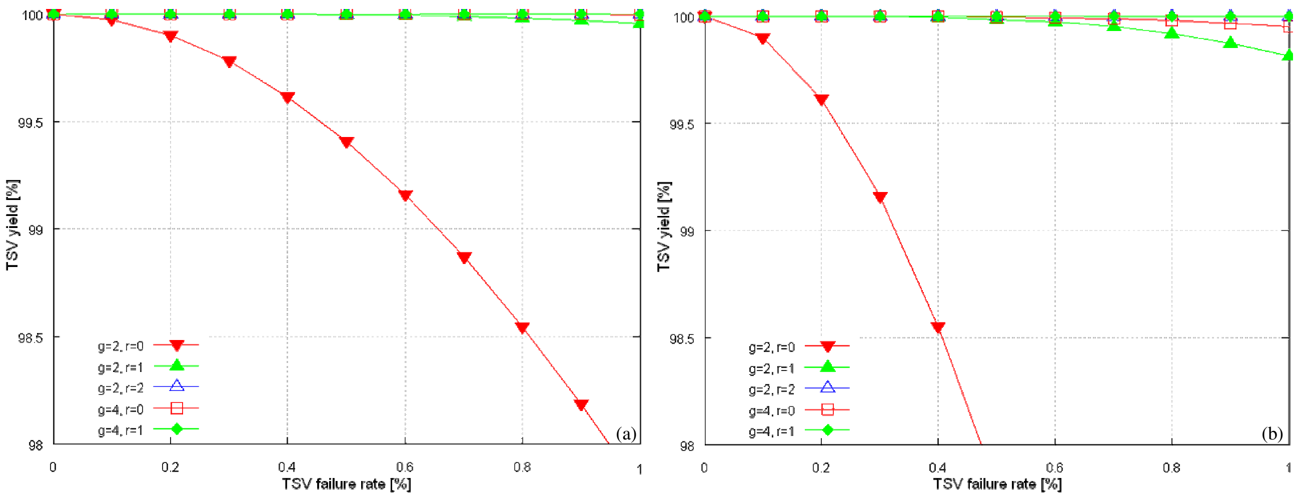


Figure V-4 TSV yield for 32-bits (a) and 64-bits (b) grouped CSLs

The results show that, even if no spares are used, high yield can be achieved by increasing the number of groups. When there are two regular TSV groups without any spares, the yield drops to *97.8%* for 32-bits links, and *92.3%* for 64-bits links. The higher yield loss of 64-bits links is due to the fact that the probability of having *32* functional TSVs per group is less that the probability of having *16* functional TSVs per group. Adding one or two spares per group increases the TSV group yield. Therefore, even if there are fewer partitions, the link yield is significantly higher than in the *no-spares* (i.e. $r=0$) case.

TSV-SnR can achieve arbitrarily high yield target by repairing faulty TSVs using functional spares, while in *CSLs* faulty TSVs are not used for data transmission, which is serialized. In the *CSL* case, the number of TSVs per chip is reduced at the expense of serialization / deserialization circuitry. The *TSV-SnR* and *CSL* techniques have been implemented for the fully synchronous seven-port router used in 3D mesh NoCs. The router has two ports for inter-die communication and the TSV repair is implemented only for them. In the next section, a comparative analysis of these techniques is presented.

5.1.1.3 Inter-die link repair trade-offs

Each link repair strategy has its advantages and limitations: spare-based repair may run out of spares, but they do not affect link performance, while *CSL*s do not use spares, but they serialize data. In this section, the two techniques are assessed with respect to their cost. In the comparative analysis, let us consider the seven-port 3D NoC router implemented in a *65nm* low-power technology process and TSVs with a *15µm* pitch.

The *32*-bits router is initially configured such that the *SnR* and *no-spares CSL* repair schemes achieve the *99.95%* target yield Y_T. In order to assess the impact of TSV scaling, it is considered that the TSV pitch is reduced to *10µm*. In this first case, the 3D NoC data size doubles (i.e. 64-bits routers), without affecting the number of (64-bits) inter-die links. In the second case, the NoC has twice as many 32-bits inter-die links and each link must achieve a *99.975%* target yield. In Figure V-5, the area overheads are summarized for ungrouped *CSL* with no spares and *SnR* configurations for the 32-bits and 64-bits routers.

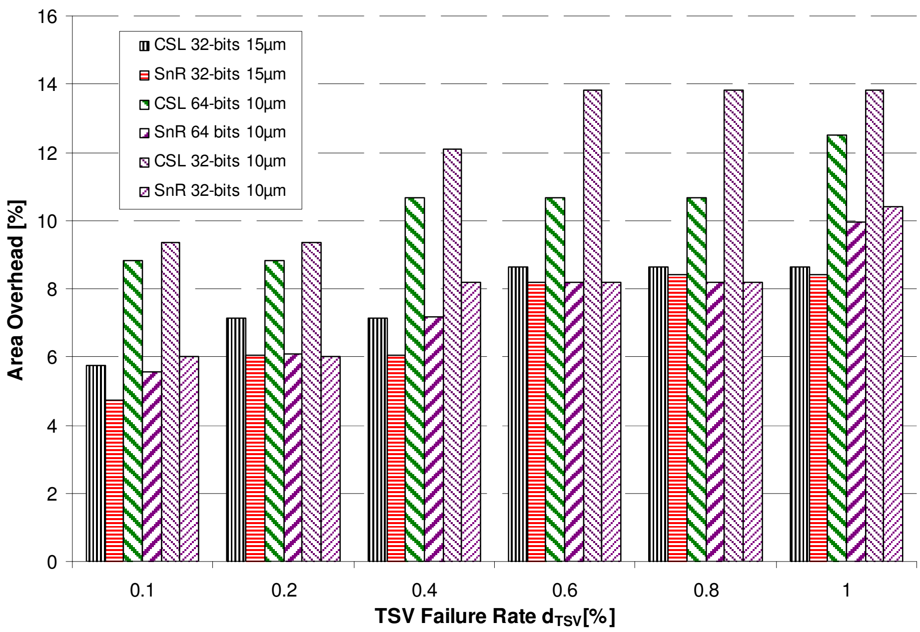


Figure V-5 Seven-port router area overheads for different TSV-SnR and CSL configurations

These results show that, in terms of area overheads, the spare-based solution is more area efficient than serialization. Increasing the TSV density by reducing p_{TSV} to $10\mu m$ may lead to wider 64-bits inter-die links or twice as many 32-bits links. If no variation of d_{TSV} with TSV density is considered, the area overheads of *SnR* remain within a *~1.5%* range, whereas the *CSL* overheads increase by up *~6%*. The variation of *CSL* area overheads are due to the complex serialization circuitry. Adding more mapping possibilities for each TSV increases the crossover switch complexity and also the number of *2:1* MUXes in the *Matrix Control Signal Selection* (MCSS) module.

Increasing the TSV failure rates d_{TSV} and the density (i.e. $p_{TSV}=10\mu m$) leads to higher area overheads for both *TSV-SnR* and *CSLs*. However, the gap between two solutions is smaller for $p_{TSV}=15\mu m$ and it decreases with d_{TSV}. In other words, when the spare TSVs footprint is significant, serialization-based repair could lead to configurations with fewer TSVs / chip and also smaller area overheads. When the TSV density doubles, the area overheads are higher for both *SnR* and *CSL*, despite the smaller TSV footprint. The area overheads of *64* bits links with $10\mu m$ pitch are lower than that of *32*-bits links with $10\mu m$ pitch TSVs.

The *TSV-SnR* costs depend on the TSV and CMOS technologies. In a *65nm* process, the repair fabric costs are determined for two TSV technologies with $10\ \mu m$ and $15\ \mu m$ pitches. For 32-bits links, the cost optimization algorithm found configurations where regular TSVs are grouped. A *TSV-SnR* area reduction of *3.75%* (i.e. less than *0.7%* of router area) was determined for $10\ \mu m$ TSV pitch and $d_{TSV}=0.4\%$. In this case, the *32* regular TSVs are split in two groups if *16* TSVs and four spare TSVs are allocated for every link. For *64* bits links, there are also cases where partitioning pays-off. The maximal area gain is *3.89%* (i.e. *~0.7%* of router area) for $d_{TSV}=1\%$ and $Y_T=99.975\%$. However, the number of extra spares per link increases from $r=5$ to $r=7$. In the analyzed technological nodes, partitioning regular TSVs in groups does not lead to significant area savings. Although the crossover switch complexity is reduced, the number of spares / chip can be significant.

The cost reduction strategy based on TSV grouping was also proposed for CSLs. Let us consider that there are up to $g=4$ groups of regular TSVs and the links are functional if at least one of these groups is functional (i.e. $g_{min}=1$ and there are up to four serialization cycles). If the $g=4$ and $g_{min}=1$ configurations do not satisfy the yield requirement then spares are allocated until the target is achieved. In the *65 nm* process, the *CSL* configurations determined for *32*-bits inter-die links with $15\mu m$ TSV pitch and $Y_T=99.95\%$ indicate an area overhead up to 5.69% for d_{TSV} up to *1%*. Reducing the TSV pitch to $10\mu m$ and widening the link to *64*-bits leads to spare-free *CSL* configurations with $g=4$ groups and area overheads up to *7.37%*. For *32*-bits links with higher yield targets and lower TSV pitch, the *CSL* grouping strategy reduces the area overheads from *13.95%* to 7.69%. Hence, grouping for CSL leads to non-negligible cost reduction, but these overheads are still above those of *TSV-SnR*. Moreover, there are cases when the *CSL* grouping strategy needs spares in order to achieve its target.

The results presented in this section have shown that it is possible to ensure high link reparability using fewer spares than in *TSV-SnR* schemes. The area overheads of the complex serialization / de-serialization circuitry often exceed the footprint of redundant TSVs in spare-based repair. Therefore, CSL may be suited only for 3D MPSoCs having a very limited number of TSVs. In 3D NoCs, an alternative to link-level fault-tolerant serialization is network configuration such that inter-die links are not used during packet routing.

5.1.2 *Network-level fault-tolerant routing algorithm*

In the network-level solution, the 3D NoC TSV yield is defined as the probability of successful repair of all inter-die links. For the *TSV fault-tolerant routing (TSV-FTR)* algorithm, this means that a master node exists for each router, and the network configuration is deadlock free. The probability of chip failure due to faulty inter-die links is very low, as solutions to the *Master Node Selection* problem exist if there are at least two nodes in neighboring layers, which are connected by a pair of functional inter-die links. In this section, the costs of *TSV-FTR* are assessed for 3D mesh NoCs.

Let us consider the seven-port router of 3D mesh NoCs, which implements the ZYX routing algorithm. In order to improve network resilience, the TSV fault-tolerant routing algorithm based on ZYX, which was presented in Chapter 4 Figure IV-23, is implemented. The hardware implementation of the ZYX routing algorithm comprises three comparators for the X, Y, and Z directions. Each of these comparators validates one of its equal (E), less (L), and greater (G) outputs. In the case of the *TSV-FTR*, two implementations are possible: low-area and high speed, which are represented in Figure V-6 (a) and (b), respectively.

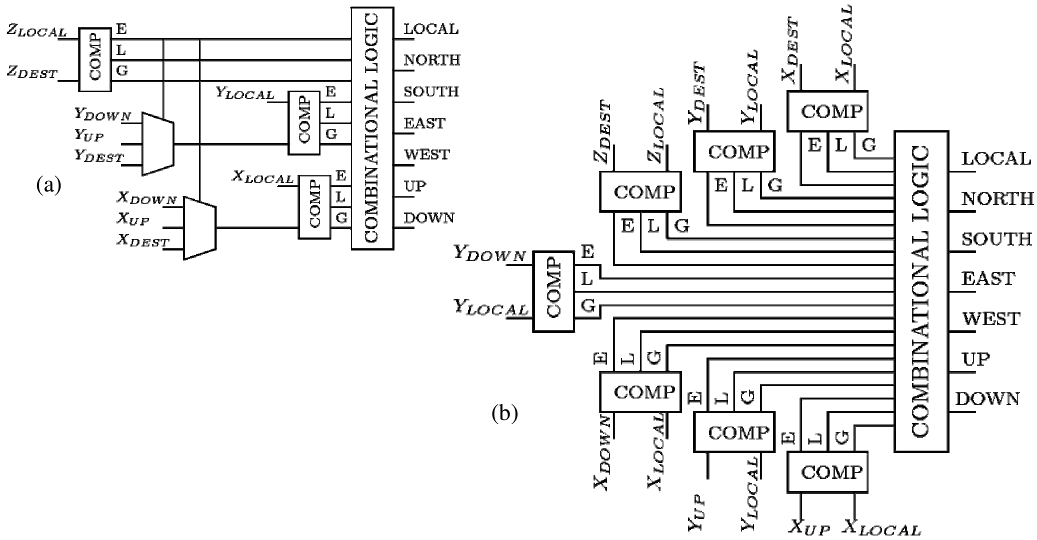


Figure V-6 Low-area (a) and high-speed(b) implementations of ZYX-based TSV-FTR

In the low-area implementation, the area overheads are negligible (i.e. less than *0.5%* in a *65 nm* low-power process). However, there is an impact on router timing, which now has a maximal frequency of *~900MHz* (i.e. $T_{clk}=1.115$ *ns*). In the high-speed implementation, the impact of the modified routing module on the router timing is minimal. The target frequency of *1 GHz* is maintained for the 65 nm target technology, but the router is more complex. The area overhead of the routing logic modules, which implement the routing function, is *~100%*, as there are *seven* comparators instead of *three*.

The routing algorithm modules account for less than *10%* of the router area. Hence, adding fault tolerant routing capabilities incurs an area / power overhead of *~7.7%/5.4%* for 32-bits routers and *6.18%/4.9%* for 64-bits routers. It can be noted that these overheads are similar to that of *TSV-SnR* with a TSV failure rate of $d_{TSV}=1\%$ and they are slightly lower than *CSL* configurations without any spares. However, the *FTR* area overheads do not depend on the TSV failure rate and pitch. In the case of high-density TSVs (i.e. TSV pitch of *10µm* or less) with low failure rates, *FTR* is more expensive than *SnR* (see Figure V-5)

In 3D NoCs, the costs of TSV yield improvement can be distributed across the data-link and network levels. The advantage of this approach is that the costs of error resilience can be reduced. In Chapter 4, a multi-layer solution for TSV yield was proposed. This strategy, which leverages *TSV-SnR*, *CSLs* and *TSV-FTR*, is assessed in the following.

5.1.3 Multi-layer TSV yield improvement

The 3D NoC TSV yield is defined as the probability of successful repair using *TSV-SnR, CSL* and the *Master Node Selection* algorithm of the fault-tolerant routing (*TSV-FTR*) strategy. In this section, the costs of the multi-layer scheme in different configurations are assessed. The comparative study also includes an assessment of the impact of single-layer (i.e. *CSL*, *TSV-FTR*), and multi-layer (i.e. *SnR-CSL* and *SnR-FTR*) solutions on the 3D NoC latency.

5.1.3.1 Area overheads

Let us consider the seven-port router of 3D mesh NoCs with the ZYX routing algorithm. In order to improve its resilience, the high-speed ZYX-based *TSV-FTR* algorithm is implemented. Although high reparability is achieved by the routing algorithm alone, spare-based repair and serialization / deserialization circuitry is added. The *CSLs* serialize data in two cycles and each link is functional if there are up to two faulty TSVs per link. In TABLE I, the area overheads of the enhanced routers are summarized for 32-/64-bits networks with TSV pitch of *15µm* and *10µm*.

TABLE I SEVEN-PORT ROUTER AREA OVERHEADS [%] FOR DIFFERENT YIELD IMPROVEMENT STRATEGIES

Data Size (bits)	p_{TSV} (µm)	FTR+ 1 Spare	FTR+ 2 Spares	FTR+CSL	FTR+ CSL+ 1 Spare
32	10µm	11.89%	13.82%	16.53%	18.65%
	15µm	12.01%	14.95%	16.53%	19.25%
64	10µm	9.62%	11.76%	16.08%	19.95%
	15µm	9.98%	12.33%	16.08%	20.1%

In the case when *TSV-FTR* is improved with spare-based repair, the probability of link failure is reduced and fewer master nodes must be determined. Hence, depending on the TSV technology, the router overheads increase from 7.7% / 6.8% to 12% / 9.9% rates for 32 / 64 data bits. Although adding more spares will not lead to significant yield gains, the probability of link failures is significantly reduced. Hence, the impact on network performance is expected to be reduced, as fewer packets will be rerouted on alternative fault-free paths.

The joint use of serialization and *TSV-FTR* has the major benefit that no spare TSVs are required. In this case, serialization reduces the link failure rate (i.e. up to one faulty TSV is repaired using serialization), but it leads to a much more complex master node selection, off-chip repair and network configuration processes. Information about serializing links, which is computed off-chip after the TSV tests, must be taken into account when master nodes are selected. In terms of area overheads, this solution is more expensive (i.e. 16%) than solutions using spares, but it does not depend on the TSV technology.

When all three solutions (i.e. spares, serialization and fault-tolerant routing) are implemented, one fault is repaired using the single spare, a second fault is repaired using serialization (i.e. $M_{MIN}=31$ for 32 bits links and $M_{MIN}=63$ for 64 bits links) and more than two faults cause link failure. Compared to other solutions in TABLE I that tolerate up to two faulty TSVs / link, the area overheads increase by up to *~8%*. Thus, this solution is less efficient.

The results above have shown that, when no spares are used, the fault tolerant routing can ensure high TSV reparability with slightly lower costs than *CSL*s (i.e. the area overheads are up to *~4%* lower). Adding spares has a small impact on the link yield and their contributions to the area overheads are non-negligible. However, having spares may improve network performance, as some TSV faults will not cause link serialization or failure. To assess the impact on network performance, a comparative study of these strategies is presented in the following.

5.1.3.2 Impact on network latency

The selection of an optimal yield improvement strategy cannot be based only on the estimated costs in terms of area, power, and number of TSVs. The impact of error resilience on network performance must also be considered. In this section, the impact of single-/multi-layer yield improvement strategies on the network latency is studied.

Let us consider a 3D mesh NoCs with error resilience implemented at data link and / or network levels. In the first case, only network-level repair is considered (i.e. TSV fault-tolerant routing *FTR*). In the second case, inter-die links are implemented as *CSLs* without any spares and no network-level repair is considered. The impact of *TSV-SnR* is not assessed, as it will not affect network latency. In the joint data link *CSL+SnR* strategy, one spare is considered for each inter-die link, while other faults are repaired using serialization. In the multi-layer scheme, the following configurations are considered: *TSV-FTR* with *TSV-SnR* (i.e. one spare per link), and no-spare CSLs (i.e. *FTR+CSL* with two tolerated faults per link). In Figure V-7, the latency overhead is shown for the 5×5×4 3D mesh topology using the TSV repair schemes above. The experimental evaluations were performed on a RTL platform with uniform traffic. In order to reduce the effects of traffic contention on performance degradation, packets is injected at very low rates such that the network is in a near *zero-load* mode.

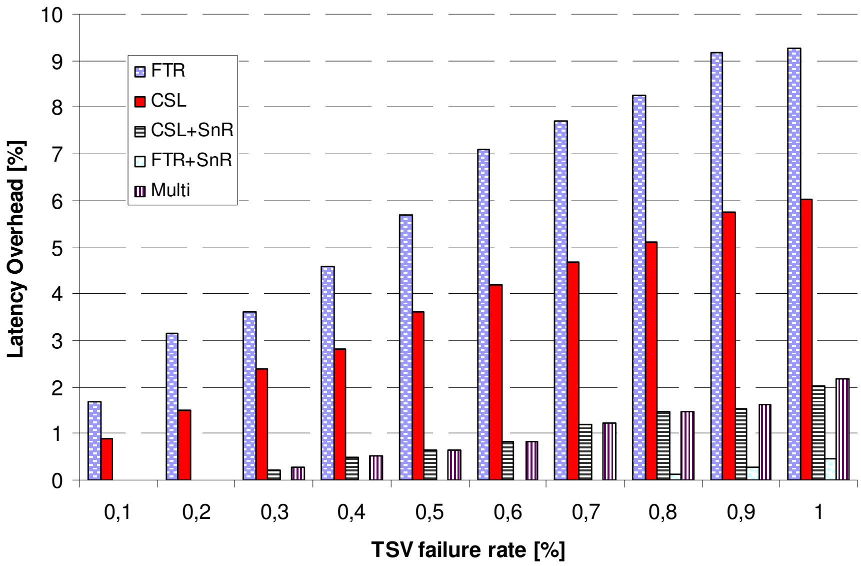


Figure V-7 Latency Overhead for 5x5x4 mesh topology using different strategies

Even for small TSV failure rates, there is a network latency penalty when *FTR* and *CSL*s are used without any spares. If there is a single faulty TSV, the link containing that wire is considered faulty in *FTR*, or it serializes data transmission. When the number of faulty TSVs increases, more links are faulty or serialize data such that for $d_{TSV}=0.1\%$ the latency increases by ~1.8% for *FTR* and 0.9% for *CSL*. The network performance degrades further for higher TSV failure rate. In the $d_{TSV}=1\%$ case, the *FTR*-protected network has a *~9.1%* latency overhead, while serialization penalizes the latency by up to *6%*.

When spares can be added, the failure / serialization probability of inter-die links drops. In the case of *CSL+SnR*, a single fault per link is tolerated without using serialization. In this case, the latency overhead is less than *1%* for lower failure rates up to *0.7%*. When *TSV-SnR* is jointly used with *FTR*, the network latency is not affected, as most faults are repaired using two spare TSVs. For higher failure rates there are more faulty inter-die links. In this case, the network latency increases by up to *0.5%*, as packets are routed around these faulty links.

In the multi-layer scheme, one fault is mitigated using spares and the second one using serialization. The link is faulty when there are more than two faults. In this case, the *FTR* algorithm routes packets around such links. For low error rates, the spare TSV is used for link repair without affecting network performance. When the number of faults increases, inter-die serialization and longer fault-free paths due to packet rerouting increase the network latency. In the multi-level scheme, the network latency goes up by *2%* for $d_{TSV}=1\%$. However, it was previously shown that the area overheads of this approach are very high.

The results presented in this section show that a multi-layer approach pays-off in some cases. While *FTR* and *CSLs* solutions ensure high yield without using spares, their impact on network performance is not always negligible. Experimental results have shown that *CSLs* can ensure high yield with less significant impact on network latency than *FTR*. In other words, if there are TSV limitations then a serialization-based solution is more efficient in terms of performance, but its costs are slightly higher.

Having spares and additional repair logic using serialization or fault-tolerant routing can improve the TSV yield, while the performance overheads are less than *2%*. The results show that the joint use of *SnR*, *CSLs* and *FTR* leads to high overheads, without any significant performance improvement compared to the *SnR+FTR* or *SnR+CSL* approaches. For the same number of TSV faults tolerated using link-level solutions, *SnR* jointly used with *FTR* ensure both high yield and negligible performance penalties (i.e. less than *1%* when two spares per link are considered).

Whether a single-layer or a multi-layer solution is used, the schemes analyzed in this section have a major limitation: they can be used only for repairing TSV faults due to manufacturing defects. In systems comprising hundreds or thousands of TSVs, strategies to tolerate in-field TSV failures are necessary. In the following, the costs of the spare-based and serialization-based equivalents of *TSV-SnR* (i.e. *IBISnR*) and *CSL* (i.e. *IBIRAS*) are evaluated.

5.1.4 ***In-field TSV failures***

Although in-field permanent faults are less likely to occur than transient faults, such failures could cause serious reliability concerns for 3D chips comprising many TSVs. In such cases, the failure of a single TSV could compromise the entire system. In this section, an assessment of the *IBIST* and *IBIRAS* (i.e. spare-/serialization-based repair) strategies is presented for 3D NoC seven-port routers.

The on-chip self-test (*Interconnect BIST*) circuitry has the advantage that, depending on the generated test patterns, it can detect both structural and parametric TSV faults. At system start-up or during system normal operation, 3D NoC inter-die links can enter an off-line TSV *test-diagnosis* phase followed by *repair*. The test patterns are generated using the *KAF* model, as described in Chapter 3. In order to reduce the self-test circuitry and test duration, TSV tests are performed using the first aggressor order. TSV tests of *n*-bits links take *16* cycles (i.e. there are two victim sets and for each set there are eight test vectors), while the *diagnosis vector* serial transmission and repair process take *n* and *4* clock cycles, respectively. Hence, the inter-die link is off-line for *n+20* clock cycles.

After TSV tests, the on-chip repair process can be performed using spares (*Interconnect Built-In Spare and Replace - IBISnR*) or adaptive serialization. Assuming a TSV failure rate during system life-time d_{TSV}, the number of spares per link *r* is determined such that TSV faults are repaired with a probability above R_T. This process is similar to the one presented in the previous section for *TSV-SnR*. In this case, d_{TSV} in Equation V-1 represents the TSV failure probability during system life-time and *Y* represented the probability that the link is reparable (i.e. at least *n* out of *n+r* TSVs are functional).

IBIRAS perform serialization-based repair when there are not enough spare TSVs. This solution is very useful, as it is very difficult to accurately predict the amount of redundancy necessary for the inter-die links to be functional during system life-time. Similarly to *CSL*s, the *IBIRAS* link is functional if at least M_{MIN} of its TSVs are functional. For a given TSV failure rate during system life-time d_{TSV}, M_{MIN} of a link with *N* regular TSVs and *R* spares is determined using equation (V-3).

In a *65nm* process, the *IBIST* and *IBIRAS* spare-based and serialization-based repair modules are implemented at the 32-/64-bits inter-die link interfaces of the seven-port router. In order to assess the impact of TSV technologies, the TSV pitch is *15µm* and *10µm*. A reparability target of 99.95% for each inter-die link is considered, without taking into account any correlations between the TSV failure rate during system life-time d_{TSV} and TSV density. Increasing the TSV density may result in 3D NoC configurations having wider links or twice as many links. In the latter case, the link reparability target is 99.975%. In Figure V-8, the area overheads of the 32-/64-bits seven-port router are represented.

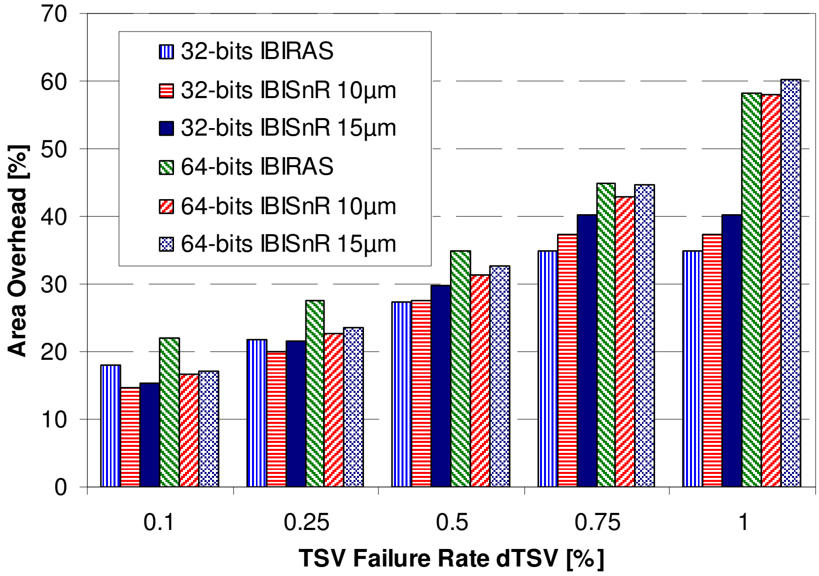


Figure V-8 Area overheads for on-chip spare-based (IBISnR) and serialization-based (IBIRAS) repair

Adding TSV test and repair capabilities during system life-time has non-negligible costs. Compared to *TSV-SnR* and *CSLs*, the router area (i.e. router logic and regular TSV footprint) overheads increase from *~10%* to more than *40%*. The extra area is due to the on-chip TSV diagnosis vector storage and serial transmission circuitry, which accounts for *~30%* of *IBIRAS* area, and the on-chip repair logic, which determines the crossover switch control signals and accounts for more than *~20%* of *IBIRAS / IBISnR* area. The area overheads due to the KAF-based TPG modules are negligible (i.e. less than *5%* of the total router area). The area overhead variation with the TSV failure rate and link size is mainly due to the increasing complexity of the crossover switches and repair logic. For *IBIRAS* and *IBISnR*, adding an extra mapping position for each regular TSV (i.e. improve link reparability) doubles the size of the repair logic. Hence, the repair logic area increases with d_{TSV} and the link width.

Unlike *TSV-SnR*, the *IBISnR* area overheads decrease with the TSV pitch for higher TSV densities and the same number of links (e.g. "*32-bits IBISnR 10µm*" and "*32-bits IBISnR 15µm*" in Figure V-8). This is caused by the reduced TSV pitch and also by the on-chip repair fabric. Although there are more spares / link in the *10µm* pitch case, the overhead due to the repair fabric is smaller than that due to the TSV pitch. Doubling the data size, as a result of TSV pitch reduction to *10µm*, results in higher overheads for all cases of TSV failure rates (see the "*32-bits IBISnR 15µm*" and "*64-bits IBISnR 10µm*" configurations above).

Although no spares are used, the costs of IBIRAS are larger than for spare-based repair. However, when the TSV failure rate increases, it can be noticed that the gap between *IBIRAS* and *IBISnR* area overheads is reducing such that 32-bits serialization repair is more efficient that spare-based repair for failure rates above *0.5%*. In the case of 64-bits links, the serialization costs are less than those of spare-based repair for failure rates above *1%*. This is explained by the fact that in the *IBIRAS* case no spares are used. Therefore, there are no area penalties for the spare TSV footprint and the *IBIST* size remains the same in the *IBIRAS* case, as no extra circuitry is necessary for spare TSVs test and diagnosis.

Choosing between *IBIRAS* and *IBSnR* cannot be based on the area overheads alone. The impact on system performance could be unacceptable for some real-time systems. In this case, serialization solutions cannot be safely used. Not allowing serialization during system life-time means that all in-field failures must be repaired using spares. In this case, accurate TSV failure models must be developed in order to ensure an optimal spare resource allocation.

5.1.5 Remarks on mitigating TSV permanent faults

The error resilience schemes have been assessed with respect to costs, overheads and impact on network performance. Each scheme comes with its own set of rules on how it must be instantiated such that the yield / reparability targets are achieved event if the number of available TSVs is limited. Using the uncorrelated fault model, the impact of TSV technologies and different trade-offs have been discussed in detail. The experimental results for TSV permanent faults are summarized as follows.

In the case when in-field failures are not considered, 3D NoCs can be repaired using spares (i.e. *TSV-SnR*), serialization (i.e. *CSL*), fault tolerant routing (i.e. *TSV-FTR*), or any combination of these solutions. For low and medium TSV failure rates, spare-based solutions have the advantage of smaller overheads and no impact on network performance. In order to minimize the number of spears / chip, it was shown that it is more efficient to allocate spares for the entire link. However, the high costs of the repair fabric may be reduced by grouping the regular TSVs in groups and allocating fewer spares / group.

For high TSV failure rates the number of spares necessary for repair increases, making this solution less attractive. Compared to CMOS technologies, the footprint of TSVs in a chip is high and, in order to reduce 3D chip costs, the number of TSVs is limited. Thus, in order to ensure high yield when no spares are available serialization and fault tolerant routing are used. Experimental results have shown that, when no spares are available, serialization solutions have a small advantage in terms of performance, but the area overheads of these solutions are similar. The situation is different when some spares are used, as *TSV-FTR* becomes more efficient both in terms of costs and performance. Using all three solution proves to have high costs with no real benefits in terms of performance.

Having to mitigate in-field failures poses several challenges: the TSVs must be tested, the repair logic must be integrated on chip. Using the *IBIST* strategy, the off-line repair scheme based on spares (*IBISnR*) and serialization (*IBIRAS*) have been analysed. Experimental results have shown that spare-based repair has some benefits in terms of costs for small and medium TSV failure rates. However, when not enough spares can be added due to high integration costs, *IBIRAS* is used. The results also showed that for highly defective TSVs with a significant TSV footprint relative to the area of the repair logic, the costs of *IBIRAS* proves to less than those of spare-based repair.

Note that the results presented in this section have been validated for the seven-port router used in 3D mesh NoCs. They are used to show how to configure these schemes and justify different instantiation choices. For example, if two spares are available per link and having two groups with one spare per group achieves the yield target then this configuration should be used. Repairing TSV permanent faults is only part of the error resilient solution. In the following section an assessment of error resilience strategies for transient faults is presented.

5.2 *Transient TSV faults in 3D NoCs*

The path reliability ρ_{path} is the probability that a flit, which traverses the path, arrives at destination fault-free. Transient faults may affect the flit during router and link traversal, causing serious reliability challenges that could lead to system failure. The reliability of an n-hops path (i.e. n routers, n-1 intra-die and inter-die links) is expressed using the router and link reliability (i.e. probability that flits traverse links and routers fault-free ρ_{link} and ρ_{router}), as shown below:

$$\rho_{path}=\rho_{link_1}\cdot\rho_{link_2}\cdot\ldots\cdot\rho_{link_{n-1}}\cdot\rho_{router_1}\cdot\rho_{router_2}\cdot\ldots\cdot\rho_{router_n} \quad \text{(V-6)}$$

In this section, an assessment of data link and network error resilience strategies for 3D NoCs is presented. Data link solutions improve path reliability by reducing the link fault probability (i.e. increase ρ_{link} in equation (V-6)), while network-level solutions improve path reliability by mitigating errors at destination. The benefits in terms of reliability and costs of selective link protection and joint data link and network error correction are discussed.

In the reliability assessments, the probability of transients on a single intra-die and inter-die wire is assumed to be up to $\varepsilon_{wire}=10^{-4}$, which is significantly more than the error rates considered in literature [BBM05, MTV05, AER10]. In the uncorrelated faults model the flit error rate $\varepsilon_{link}=1-(1-\varepsilon_{wire})^n$ is proportional with the link size n. Thus, the link reliability is $\rho_{link}=99.68\%$ for 32-bits flits, and $\rho_{link}=99.36\%$ for 64-bits flits. Because faults are uncorrelated, most errors are due to single faults, which occur with a *0.31%* for 32-bits links and *0.63%* for 64-bits links, rather than multiple faults (e.g. double faults occur with a *0.0004%* probability for 32-bits links and *0.002%* for 64-bits links).

The router reliability ρ_{router} quantifies the probability that a single flit traverses the router (i.e. input/output FIFOs, internal crossover switches) fault-free and not the probability that the router functionality is affected by transients. Errors that affect routing logic, internal state registers (e.g. FIFO pointers, FSM state register, etc), and other combinational logic are not addressed in this thesis.

Although the link reliability seems high enough, paths comprise many links and routers. For example, in the case of *12*-hops paths, even if a $\rho_{router}=99.99\%$ router reliability is assumed, there is a *96.23%* probability that flits arrive fault-free for 32-bits flits, and *92.6%* for 64-bits flits. When extended to packets with eight flits, communication reliability drops to *72%* and *54%*, respectively. In other words, ~¼ packets traversing the 32-bits path contain errors and almost ½ of packets traversing the 64-bits path arrive with errors. The goal of error resilience techniques is to increase communication reliability to acceptable levels, which are often above *99.99%*.

5.2.1 *Data link error resilience*

The data link error resilience schemes improve communication reliability by improving link reliability ρ_{link}. In the following, the costs of error control strategies (i.e. *Forward Error Correction*, *Automatic Retransmission Query* and *Hybrid Error Correction and Retransmission*) are estimated for the seven-port routers of 3D mesh NoCs.

5.2.1.1 *Reliability assessments*

In choosing a link protection strategy, one faces the problem of selecting an appropriate coding strategy such that the link reliability satisfies the targets. In this section, the reliability of each error control scheme is assessed for different coding strategies. An error detection and correction code selection strategy based on interleaving is also presented for the cases where multi error mitigation is necessary.

For *Forward Error Correction* (FEC) schemes, link reliability ρ^{FEC} depends on the code error correction capabilities. The probability ρ^{FEC} that flits correctly traverse links is given by the probability that the transient faults pattern is correctable. For k-bits flits (i.e. data and error check bits), if i errors occur with the probability ε_i and corrected with probability C_i then ρ^{FEC} is given by:

$$\rho^{FEC} = \varepsilon_0 + \sum_{i=1}^{k} C_i \cdot \varepsilon_i, \quad \varepsilon_i = \binom{k}{i} \cdot \varepsilon_{wire}{}^{i} \cdot (1-\varepsilon_{wire})^{k-i} \qquad \text{(V-7)}$$

Equation (V-7) shows that the link reliability is the probability that no error occur or that error that occur are corrected. FEC-protected link reliability can be improved by implementing codes with multiple error correction capabilities. For on-chip communication, most *FEC* schemes rely on *Single Error Correction* (SEC) codes like Hamming. When multiple error correction is desired, SEC codes are interleaved: data bits are split in two or more disjoint groups and each of these groups is encoded using a SEC code. Having more groups increases the correction capabilities of the code: when data is split in m groups, multiple error patters of up to m errors can be corrected if the errors are distributed such that there is at most one error per group. Thus, $C_1=1$ for $C_i \neq 0$, for $m \geq i > 1$, and $C_i=0$, for $i>m$.

Let us consider a *12*-hops path where all links are protected using FEC with interleaved Hamming codes. In Figure V-9, the cumulated reliability of 32-bits and 64-bits links along the path is represented. The link reliability is determined using equation (V-7) with different correction probabilities, depending on the number of interleaved SEC codes.

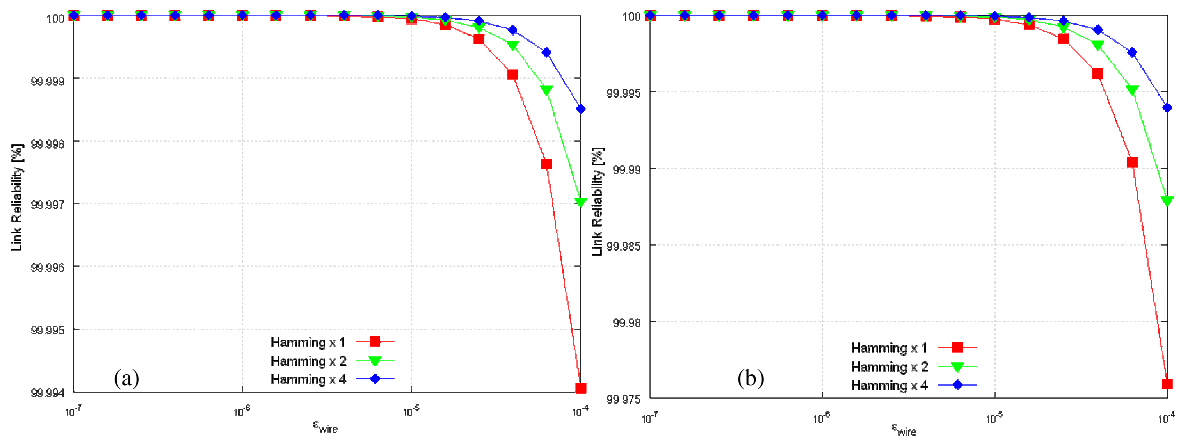


Figure V-9 Link-component of 12-hops path reliability for 32 (a) and 64 (b) bits FEC-protected links

Hamming SEC codes improve the reliability of each link such that *64*-bits flits traverse the *12* hops fault-free with a probability above *99.975%*, while *32*-bits flits traverse path with a probability above *99.994%*. When interleaving is considered, the link reliability increases, as up to four errors per link (i.e. "*Hamming*×*4*" in Figure V-9) can be corrected. When interleaved codes are used, the number of wires per link (i.e. codeword size) k increases. Although the probability of single errors ε_1 increases, the probability of having multiple errors remains very low. Therefore, the reliability gains by SEC interleaving are relatively small: up to *0.005%* for 32-bits links and up to *0.02%* for 64-bits links.

The reliability of *Automatic Retransmission Query* (ARQ) scheme ρ^{ARQ} is directly affected by the code error detection capabilities ε and residual error rate rez (i.e. probability that errors are not detected). Let D_i represent the probability that i errors are detected for the k-bits encoded flit. Using D_i, the error detection capabilities ε and its residual error rate rez are expressed as:

$$\varepsilon = \sum_{i=1}^{k} D_i \cdot \varepsilon_i \qquad \text{(V-8)}$$

$$rez = \sum_{i=1}^{k} (1 - D_i) \cdot \varepsilon_i \qquad \text{(V-9)}$$

In the *go-back-N* retransmission scheme, successive transmissions are not independent events and ρ^{ARQ} depends on the successful transmission of *N-1* previous flits, where N is the retransmission buffer size. For $N=4$, let us consider three consecutively sent flits F_1, F_2 and F_3. If a retransmission request is made for F_1 then flits F_2 and F_3 are discarded even if they are received correctly. The probability that no errors are detected for F_1 depends on ε and n_{RT}, where n_{RT} is the maximum number of consecutive retransmission requests for the same flit. As mentioned in the previous chapter, the *retransmission logic* is simplified by allowing only one retransmission (i.e. $n_{RT}=1$). In equation (V-10), the probability that F_1, F_2 and F_3 correctly traverse the *ARQ*-protected link is given.

$$\rho_i^{ARQ} = (1-\varepsilon)^i \cdot (1 + i \cdot \varepsilon) - rez \quad 1 \le i \le 3 \qquad \text{(V-10)}$$

From this equation it is clear that the link reliability depends on the error detection probabilities. For the *12*-hops path, let us consider perfect routers and *ARQ*-protected links. The link component of path reliability (i.e. $\rho_{link1} \cdot \ldots \cdot \rho_{link11}$) is determined for 32-bits and 64-bits flits using different parity encoding strategies: simple parity, interleaved parity on two and four groups and *CRC-8* using the generator polynomial $G(X)=1+X^4+X^5+X^8$. In Figure V-10, the cumulated link reliability of 32-bits and 64-bits links is represented.

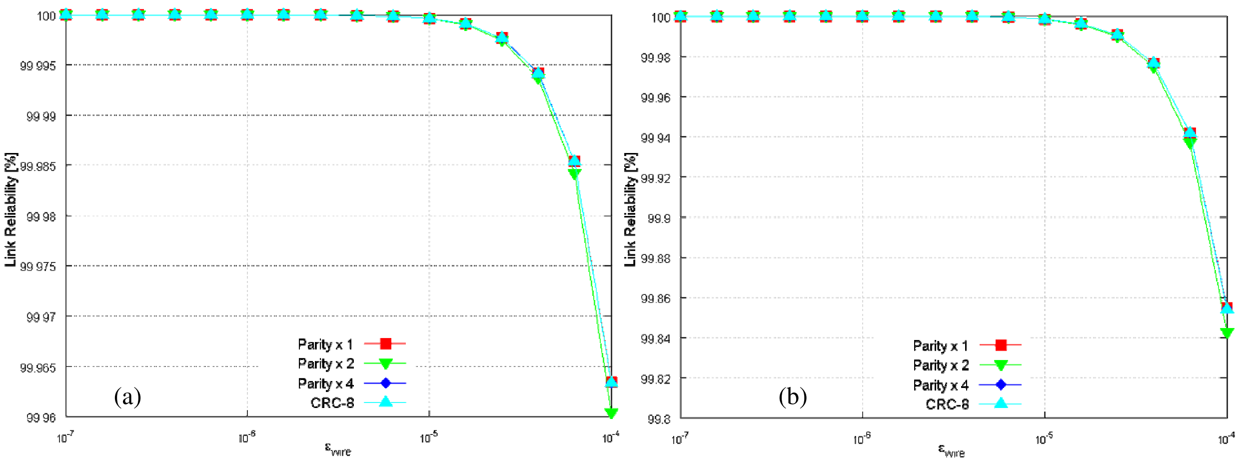


Figure V-10 Link-component of 12-hops path reliability for 32 (a) and 64 (b) bits ARQ-protected links

Compared to the un-protected link case, the *32*-bits and *64*-bits link reliability increases from *99.63%* to *99.96%* and from *99.36%* to *99.84%*, respectively. Compared to *FEC*, the lower reliability of *ARQ* is mainly due to the *go-back-N* retransmission mechanism. Even when a flit arrives correctly, it is discarded and retransmitted if the flit before was erroneous. In the un-correlated fault model multiple errors per link are unlikely. Therefore, most errors are detected and there are no major reliability differences for different coding schemes.

The reliability of *hybrid error-correction and retransmission* (HYB) schemes depends on the coding correction / detection capabilities and its residual error rate *rez* (i.e. probability that errors go undetected). For *k* bits flits, let D_i represent the probability that *i-out-of-k* errors are detectable and C_i represent the probability that *i-out-of-k* errors are correctable. Note that for most error correction codes the detection capabilities exceed the correction ones and $D_i \geq C_i$ for $i \geq 1$. In HYB, retransmissions requests are made only for flits having uncorrectable, but detectable errors. The probability of detectable, but uncorrectable errors ε_R is expressed as:

$$\varepsilon_R = \sum_{i=1}^{k} D_i \cdot (1 - C_i) \cdot \varepsilon_i \qquad \text{(V-11)}$$

In the *go-back-N* retransmission strategy, the reliability of flit transmission depends on the transmission of previous *N-1* flits, where *N* is the retransmission buffer size. For *N=5*, let us consider four consecutively sent flits F_1, F_2, F_3 and F_4. If a retransmission request is made for F_1 then flits F_2, F_3 and F_4 are discarded even if they are correctly received. In this case, the probability that flit F_1 arrives error-free at the link downstream interface depends on ε_R and n_{RT} (i.e. the maximum number of retransmission requests). The probabilities that F_2-F_4 arrive error-free depend on the previous flits. Using the same reasoning as for *ARQ*, the *HYB* link reliability is given in equation (V-12) for each F_i with $1 \leq i \leq 4$.

$$\rho_i^{HYB} = (1 - \varepsilon_R)^i \cdot (1 + i \cdot \varepsilon_R) \quad 1 \leq i \leq 4 \qquad \text{(V-12)}$$

HYB schemes rely on *Single Error Correction Double Error Detection* (SECDED) codes like extended Hamming codes (i.e. Hamming SEC with a parity bit for double error detection) or Hsiao codes. The multiple error correction issue is addressed by *SECDED* code interleaving. Even when no interleaving is considered, the HYB-protected link reliability is significantly higher than for *FEC* and *ARQ* schemes (i.e. >> 99.99% even for $\varepsilon_{wire}=10^{-4}$). In HYB links, single errors are corrected, double errors cause retransmission, and triple errors are very unlikely. Moreover, in case of a retransmission, it is very unlikely that double errors affect consecutive transmission cycles. For the *12*-hops path, the cumulated reliability of all links exceeds *99.9999%*, even for 64-bits links.

The results above have shown that link-level protection is a powerful strategy for improving path reliability. Protecting each link of the 3D NoC may prove expensive and, in some cases, un-necessary. In the following, the reliability of selective link protection strategies is evaluated.

5.2.1.2 Selective inter-die link protection

In 3D NoCs, let ρ_h and ρ_v represent the probabilities that flits traverse unprotected intra-die / inter-die links fault-free, respectively. For paths comprising n_h intra-die links and n_v inter-die links, the cumulated link reliability along the path is expressed as:

$$\rho_{path}^{links} = \rho_h^{\,n_h} \cdot \rho_v^{\,n_v} \tag{V-13}$$

The link-level error resilience schemes improve the 3D NoC communication reliability by reducing the probability of transient errors on intra-die and inter-die link traversal. The probability that flits traverse without errors protected intra-die / inter-die links (i.e. ρ_h' / ρ_v') is given by equations (V-7, V-10, V-12). Using these equations, the path reliability can be expressed as:

$$\rho_{path}^{links} = \rho_n{'}^{n_h} \cdot \rho_v{'}^{n_v} \tag{V-14}$$

Protecting all links in 3D NoCs might be prohibitively expensive. In order to reduce these costs, the selection protection scheme is used. Therefore, error resilience is implemented only for inter-die links. The probability that flits correctly traverse the selective protected path is given in equation (V-15).

$$\rho_{path}^{links''} = \rho_n^{\,n_h} \cdot \rho_v{'}^{n_v} \tag{V-15}$$

There is a reliability loss when only inter-die links are protected, as intra-die links are not protected. Errors on intra-die link propagate to destination, causing a reliability loss. Moreover, path reliability cannot be improved above a level determined by the intra-die error rates.

In order to assess the impact of *all links* and *inter-die links* protection strategies, let us consider a 12-hops path with $n_h=9$ intra-die links and $n_v=2$ inter-die links. In Figure V-11, the path reliability is represented for *32*-bits (a) and *64*-bits links (b) when *all* links and selective inter-die link protection strategies are used. Note that a wire error rate of 10^{-6} is assumed for intra-die links.

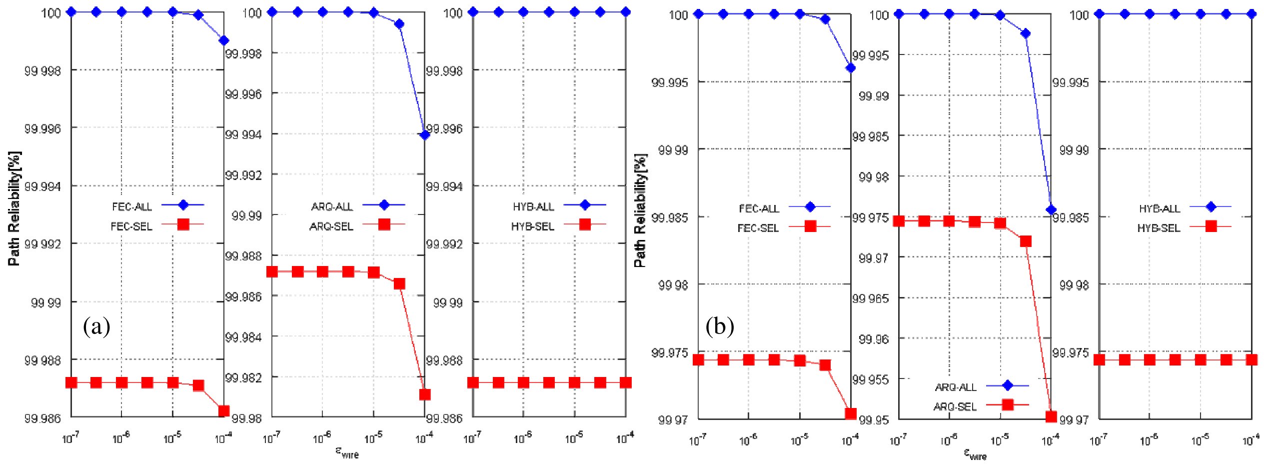


Figure V-11 Path reliability with error resilient inter-die links (i.e. selective inter-die protection SEL) and full-link protection (ALL) for an intra-die wire failure rate of $\varepsilon_{wire}=10^{-6}$

Intra-die errors will cumulate along the path, resulting in some reliability penalty: *0.012%* for 32-bits links and *0.025%* for 64-bits links. For Hamming SEC protected links (i.e. FEC), the path reliability drops when the inter-die wire error rate is 10^{-4}. In this case, the path reliability is dominated by transients on inter-die link. The reliability gap is wider for CRC-based ARQ: *99.995%* for all link protection and *99.97%* for inter-die link protection. In the case of Hamming SECDED *HYB* scheme, there is no visible drop for higher inter-die error rates: all inter-die errors are mitigated at link-level and they do not propagate to the destination.

For higher intra-die error rates, the reliability penalty increases, since intra-die faults are more likely to occur. In the case where intra-die wires error rates is $\varepsilon_{wire}=10^{-4}$, the link component of path reliability will decrease from *~99.9%*, in the case of all link protection, to *~97.5%* for 32-bits flits and *~94.5%* for 64-bits flits. Therefore, reliable communication by selective inter-die link protection cannot be guaranteed in this case.

The selective link-level protection is an efficient strategy for networks with less strict reliability targets. This strategy can also be used in the case when non-protected links have low-medium error rates (i.e. $\varepsilon_{wire} < 10^{-6}$). In order to assess the benefits of selective inter-die link protection in 3D NoCs, a comparative study with respect to the costs in terms of area and power, and impact on network performance, is presented in the following.

5.2.1.3 *Area and power overheads*

The link-level error resilience schemes using the *all* links and the *selective inter-die* link protection strategies are implemented for seven-port routers in a *65nm* technology. In order to maintain the target *1GHz* clock frequency, intermediate retiming stages are inserted between the *PHY*, the encoding, and the error detection / correction modules.

In *FEC*-protected links, a single retiming stage is necessary between the encoder and the decoder. Hence, for 32-bits and 64-bits links, data encoding and *PHY* traversal take one clock cycle, while error detection and correction is performed in another cycle. An intermediate retiming stage is also inserted for ARQ-protected links. In the first cycle data is encoded and *PHY* is traversed, while error detection and retransmission request is performed in the second cycle. In the case of *HYB* schemes, it is not possible to detect, correct the error and sent the retransmission request within a clock cycle. Hence, in the first transmission cycle the data bits are encoded and the physical link is traversed, in the second cycle errors are detected, and in the third cycle errors are corrected and retransmission requests are sent.

Adding intermediate retiming stages between the router interfaces raises the issue of input buffer overflow: when the receiver interface signals that the input buffer is full, the transmitter interface could have sent a new flit that cannot be stored in the input *FIFO*. This problem is addressed by adding q extra positions to the input buffer such that if the buffer has n positions, its full signal will be active when at least n-q of its positions are occupied. In the cases above, a single position is added for each input buffer of a protected link.

In TABLE II, the area and power overheads of 32-bits and 64-bits routers (without the TSV footprint) are summarized for all links and selective inter-die link protection strategies using *FEC* (i.e. interleaved Hamming SEC codes), *ARQ* (i.e. CRC codes and interleaved parity) and *HYB* (i.e. interleaved Hamming SECDED codes) error control schemes.

TABLE II AREA/POWER OVERHEADS [%] FOR 32/64 BITS SEVEN-PORT ROUTER WITH LINK-LEVEL PROTECTION ON ALL LINKS AND INTER-DIE LINKS ONLY

Error Control Strategy	32 Bits				64 Bits			
	All links		Inter-die links only		All links		Inter-die links only	
	Area (%)	*Power (%)*	*Area (%)*	*Power (%)*	*Area (%)*	*Power (%)*	*Area (%)*	*Power (%)*
FEC SEC	25.90	35.78	8.46	11.22	26.7	35.32	8.73	11.51
FEC SEC×2	26.98	38.36	9.09	12.12	27.39	37.43	8.9	12.02
FEC SEC×4	28.11	43.22	9.41	13.23	27.95	40.34	9.16	13.31
ARQ CRC-5	36.65	46.96	10.2	13.45	39.9	48.6	11.25	14.17
ARQ CRC-8	38.17	48.44	10.71	13.50	44.52	48.78	11.13	14.26
ARQ PARITY×4	42.85	57.21	12.16	16.44	45.64	43.07	16.35	10.63
HYB SECDED	77.15	95.49	21.34	27.58	83.56	95.35	22.98	27.97
HYB SECDED×2	77.16	103.11	21.27	29.32	81.57	99.03	22.62	29.31

The results show that inter-die link protection is ~*3*× less expensive than all links protection, as error protection is implemented for two of the six adjacent links. Improving the link reliability by using *FEC* schemes with interleaved Hamming codes (i.e. SEC×2 for two groups and SEC×4 for four groups) increases the router area and dissipated power, as the complexity of the coding scheme increases. Although the area overhead increase from one group to four interleaved groups is less than *1%* for 32-bits routers and *2%* 64-bits routers, the code word size increases from *38* to *48* and *71* to *84*, respectively. Hence, when the TSV footprint is considered (i.e. *10μm* pitch), the area overheads of inter-die link protection are between *11.5%* and *17%* for 32-bits routers, and between 9% and *13%* for 64-bits routers.

In *ARQ* schemes the retransmission buffer size is $N=3$ flits, as a single retiming stage is inserted between the *PHY* and decoder. Although *ARQ* uses simpler codes than *FEC*, its slightly larger overheads are due to the retransmission buffer. The main advantage of *ARQ* over *FEC* is that fewer extra wires are needed for error control bits transmission. The number of extra TSVs per link is *4* TSVs for interleaved parity on four data groups (i.e. PARITY×4), *5* TSVs for CRC-5 (i.e. $G(X)=1+X+X^3+X^5$) and *8* TSVs for CRC-8 (i.e. $G(X)=1+X^3+X^5+X^6+X^8$), for both 32 and 64 bits routers. In a $10\mu m$ pitch TSV technology with interleaved parity on four groups, the area overheads are *~10%* for 32-bits links and *13%* for 64-bits links. Interleaved parity encoding is less complex than for CRCs, but the area overheads are slightly larger, while the power overheads are smaller for 64-bits routers. Although the ARQ costs are slightly larger than those of FEC, the advantage of this solution is that fewer TSVs are necessary for error check bits transmission.

Hybrid error correction and retransmission schemes offer the best link-level reliability, but their area and power overheads are significantly higher. Because error detection / correction and the retransmission request cannot be done in a single cycle, an intermediate retiming stage is considered between the detection and correction stage. Hence, the retransmission FIFO size is $N=4$ and two extra positions are added in the router's input buffers in order to prevent data loss. Because double errors must be correctly identified, Hamming SECDED codes must be used. Interleaving increases the error correction and detection capabilities, but it also increases the coding and decoding complexity. Moreover, the number of wires used for error check bits transmission is larger. In the case of 32 bits links, the number of TSVs increases from 7 to 12, when the data bits are encoded using two interleaved groups.

5.2.1.4 Impact on network latency

Network latency is a performance metric that indicates the time needed by a packet to traverse the network *flit-by-flit*. When all links are protected, the latency of each traversed link increases by one clock cycle even if no errors are detected. In the case of inter-die link protection, the packet latency increases only for the traversed inter-die links. When transient errors are detected, the latency of *ARQ*-protected links increases, as flit retransmission takes up to four cycles.

In the case of 32-bits and 64-bits FEC-protected links, error detection and correction is done in a single cycle. Hence, the link latency is not affected by the error rate ε_{wire}. For ARQ-protected links, flits affected by transients traverse the link in four cycles instead of two. The latency of HYB-protected links depends on whether the error is correctable or not. If the error is correctable then the flit latency is three cycles. When a retransmission request is sent, the flit latency is five cycles.

In the network latency estimations, errors are injected with the same error rate on protected intra-die and inter-die links. Note that in the case of selective inter-die protection, no intra-die errors are considered, as there are no mechanisms to mitigate them. Let us consider the 4×4×4 3D mesh NoC on which eight-flit packets are randomly transmitted with a uniform distribution. In Figure V-12, the average network latency of the 32-bits and 64-bits 3D NoC is represented of all-links and selective link protection using *FEC*, *ARQ*, and *HYB*.

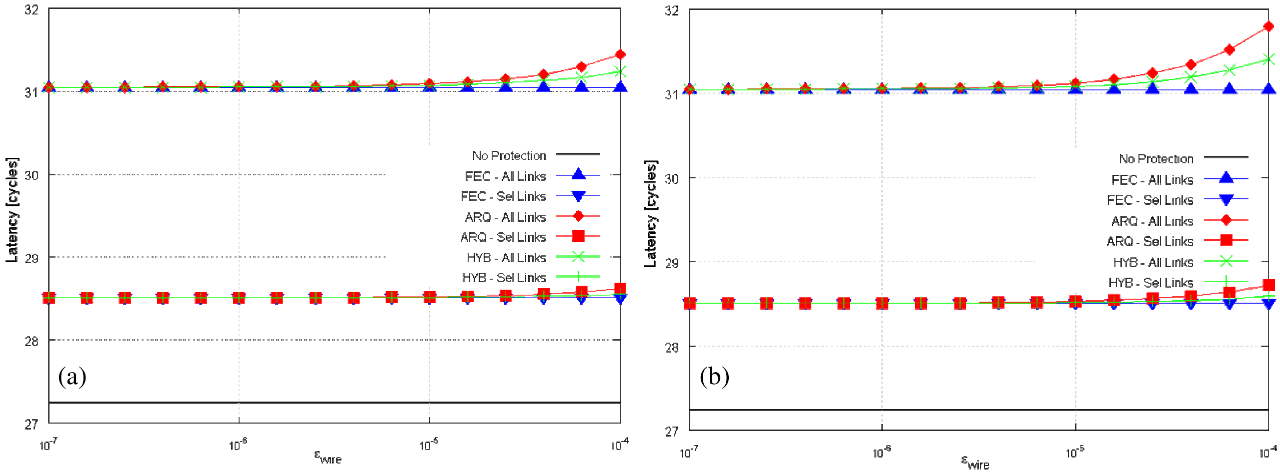


Figure V-12 Average network Latency for 32-bits (a)and 64-bits (b) 4×4×4 3D NoCs

The results above show that the performance penalty of selective inter-die link protection is significantly smaller than that of all link protection. In the case of low error rates, the network latency increases by more than one cycle for inter-die link protection and by almost four cycles for all link protection (i.e. *~15%* latency overhead). Increasing the error rates impacts the network latency, as error recovery may take up to three cycles per link.

As mentioned before, there is no impact of ε_{wire} on the network latency when links are protected using *FEC*. However, interleaved Hamming SEC codes are used for higher values of ε_{wire}. For *ARQ*, the impact of link-level flit retransmission on the latency is noticeable for error rates higher than 10^{-5}. In *HYB* protected links with Hamming SECDED codes, when single errors are detected the network latency increases, as error correction is performed in an extra cycle.

For the un-correlated transient fault model, the impact of *HYB* on network latency is mainly due to the extra cycle needed for error correction. In the evaluations above, the flit size is also important, as the flit error rate depends on its size. Therefore, the performance penalties are slightly higher for 64-bits links, as transients are more likely to occur.

The results in this section have shown that reliable communication can be achieved by means of link-level error resilience. Protecting all links of a network may prove to be too expensive and leads to high overheads and performance penalties (i.e. *~15%* latency overhead). In the selective link protection strategy, path reliability is traded for lower area / power overheads and network performance. Experimental results have shown that this strategy pays-off when intra-die links and router are less prone to errors. However, transients on links can also be mitigated at network level. In the following section an assessment of network-level solutions is presented.

5.2.2 Network-level error resilience

Transient errors on the *PHY* can be mitigated at network-level by encoding individual flits at the source node and correcting errors at destination (i.e. *Network-level Forward Error Correction NL-FEC*). Unlike in link-level solutions, flits traverse a series of intra-die / inter-die links and routers before errors are detected and corrected. Therefore, *NL-FEC* mitigates transients that affect flits during link and router traversal.

5.2.2.1 *Reliability assessments*

For *NL-FEC* protection, let us consider an *n*-hops path that comprises *n* routers and *n-1* intra-/inter-die links. The probability that a flit correctly arrives at destination $\rho^{NL\text{-}FEC}$ can be expressed using the probability that it traverses routers and links fault-free (i.e. ρ_r, ρ_l):

$$\rho_{flit}^{NL-FEC} = \rho_l^{\,n+1} \cdot \rho_r^{\,n} + \sum_{i=1}^{e} \left[\binom{n+1}{i} \cdot (1-\rho_l)^i \cdot \rho_l^{\,n+1-i} \cdot \rho_r^{\,n} + \binom{n}{i} \cdot (1-\rho_r)^i \cdot \rho_r^{\,n-i} \cdot \rho_l^{\,n+1} \right] \qquad \text{(V-16)}$$

In equation (V-16), *e* represents the number of correctable errors per flit. Since errors are corrected only at destination, they may cumulate along the source-destination path. Hence, error resilience is improved by implementing multiple error correction codes based on interleaved SEC codes.

Let us consider a *12*-hops path with a single wire error rate $\varepsilon_{wire} \leq 10^{-4}$ and a router reliability ρ_r=*99.99%*. (i.e. the probability that a single flit traverses the router fault-free). In Figure V-13, the *NL-FEC* protected reliability of a 32-bits and 64-bits *12*-hops path is represented.

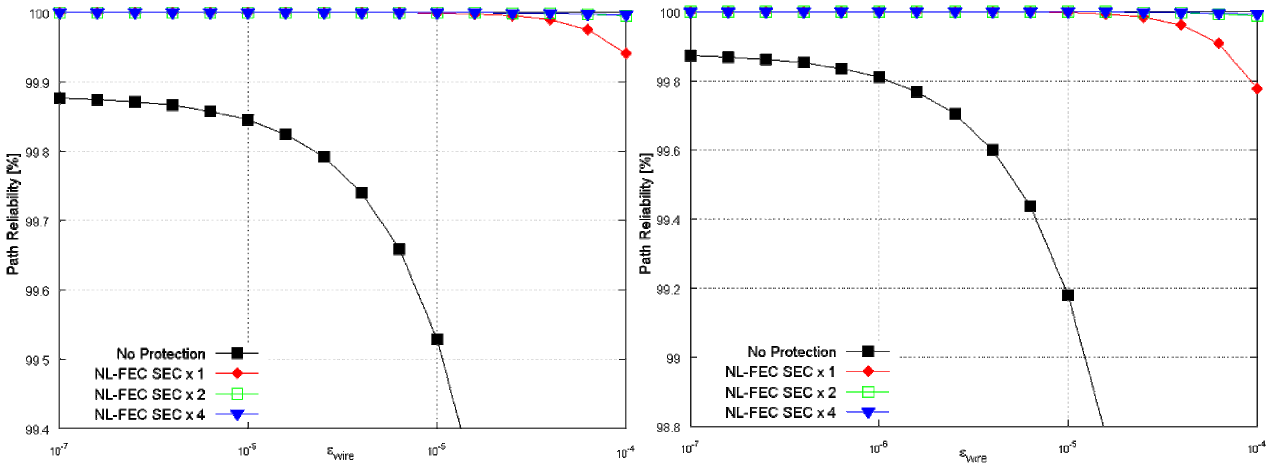


Figure V-13 Path reliability for 32-bits and 64-bits networks using NL-FEC with interleaved Hamming

When the path is not protected, the probability that flits traverse it correctly drops to less than 90% for $\varepsilon_{wire}=10^{-4}$. When flits are encoded using a SEC code, the path reliability is above *99.95%* for *32*-bits flits and *99.8%* for 64-bits flits. SEC code interleaving increases the error correction capabilities such that up to four errors per flits can be corrected. In this case, the path reliability is above *99.999%* for 32-/64-bits links.

Network-level protection ensures higher reliability by mitigating transients both on links and routers, while link-level solutions correct only link errors. For a *12*-hops path with *99.99%* router reliability, link-level and network-level *FEC* solutions are implemented. Error correction modules are implemented at destination for NL-FEC, while LL-FEC is implemented for *all* links in the path. In Figure V-14, the path reliability is represented for different interleaved Hamming SEC codes.

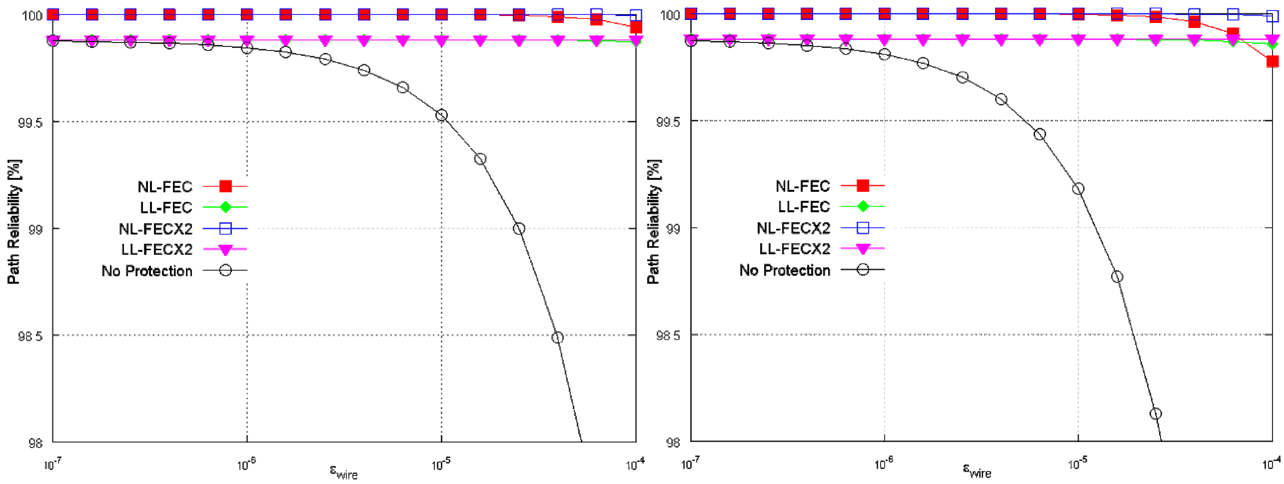


Figure V-14 Path reliability for 32-bits and 64-bits networks using link-level and network-level FEC

Although link-level protection improves path reliability, it cannot exceed the *99.9%* boundary set by the router reliability. Network-level protection ensures higher path reliability, as errors that cumulate on routers and links are corrected. The probability of cumulating multiple un-correctable faults increases for wider links and high wire error rate. In this case, the *NL-FEC* path reliability drops while *LL-FEC* path reliability remains high, as most errors are corrected. It can be noted that the *NL-FEC* path reliability is lower than *LL-FEC* path reliability for *64*-bits flits and error rates higher than $\varepsilon_{wire} \geq 8 \cdot 10^{-5}$. The NL-/LL-FEC reliability breakeven point shift to the left when link failures are more likely than router failures (i.e. router reliability drops). Interleaving SEC codes enable *NL-FEC* scheme to correct up to two errors per flit. In this case, the path reliability is above *99.999%* even for $\varepsilon_{wire}=10^{-4}$. At link-level, interleaved SEC codes can correct double errors such that there is a reliability gain. However, this gain is relatively small, since the probability of double errors is the uncorrelated faults model is low.

5.2.2.2 Area and power overhead estimations

The *NL-FEC* encoding / decoding modules are added to the *LOCAL* input / output ports of the 32-bits and 64-bits 3D NoC seven-port routers. In TABLE III, the area and power overheads are summarized for different Hamming SEC coding schemes.

TABLE III OVERHEADS OF NL-FEC PROTECTED SEVEN-PORT ROUTERS

Error correction scheme	**32-bits**		**64-bits**	
	Area (%)	**Power (%)**	**Area (%)**	**Power (%)**
Hamming SEC × 1	19.81	15.55	18.84	9.75
Hamming SEC × 2	33.61	30.61	24.93	16.62
Hamming SEC × 4	49.88	48.78	37.72	28.98

The router area and power overheads are mainly due to input / output buffers, which account for *~90%* of router area. When flits are encoded, the buffer size increases, as the error check bits are appended to the original flit. The complexity of encoding / decoding modules increases with data size. However, the area / power overheads decrease, as the error control bits / data bits ratio decreases: 6 error control bits for 32 data bits and 7 error control bits for 64 data bits.

Interleaving SEC codes leads to larger flits (i.e. more error check bits are necessary to protect data) that have a major impact on the area / power overheads of *NL-FEC* protected routers. When there are two and four interleaved data groups, the number of control bits increases from *6* to *10* and *16* for 32-bits networks, and from *7* to *12* and *20* for 64-bits networks. Thus, the number of intra-die / inter-die wires increases in order to allow data and error check bit parallel transmission. Compared to link-level *FEC*, *NL-FEC* is more expensive when interleaved codes are used (i.e. *LL-FEC* area / power overheads are up to *28% / 40%*).

In the *65nm* process, flit encoding and error detection / correction can be performed within one clock for 32-bits and 64-bits *NL-FEC*. Hence, there is no impact on network performance. This result, together with the area / power overheads and reliability assessments above show that *NL-FEC* is more efficient in the case when network performance is critical and multiple transients do not cumulate along the path.

5.2.3 Multi-layer error resilience

A limitation of network-level *FEC* is that faults cumulate along the path, leading to uncorrectable error patterns. In the multi-layer approach, network reliability is improved by adding link-level *Forward Error Correction* capabilities. Using the flit error control bits appended at the source node, correction is possible at link-level by adding correction stages.

5.2.3.1 Reliability assessments

Let us consider a path having *n* routers (i.e. *n-1* identical links) and *c* intermediate correction stages. The path reliability (i.e. the probability that flits correctly arrive at destination) is given by the probability that all errors are corrected by the intermediate correction stages or at destination. Assuming that there are $s=c+1$ sections with $n_1 \dots n_s$ routers each, the path reliability is given by:

$$\begin{aligned} \rho_{flit}^{FEC} &= \prod_{i=1}^{s} \rho_{path}(n_i) \\ \rho_{path}(n_i) &= \rho_{link}{}^{n_i+1} \cdot \rho_{router}{}^{n_i} + \sum_{j=1}^{e} \left[\binom{n_i+1}{j} \cdot (1-\rho_{link})^j \cdot \rho_{link}{}^{n_i+1-j} \cdot \rho_{router}{}^{n_i} + \binom{n_i}{j} \cdot (1-\rho_{router})^j \cdot \rho_{router}{}^{n_i-j} \cdot \rho_{link}{}^{n_i+1} \right] \end{aligned} \quad \text{(V-17)}$$

Equation (V-17) shows that network reliability (i.e. the minimal path reliability) depends on the code correction capabilities and the path length between correction stages. In the multi-layer scheme, Hamming codes are used for single error correction (SEC). However, when multiple error correction is required, SEC codes are interleaved.

For a *12*-hops path with 32-/64-bits links and a router reliability $\rho_r=99.99\%$, let us consider that correction stages are inserted every two, three, four and six hops. In Figure V-15, the reliability of *NL-FEC* schemes with up to six intermediate correction stages and with interleaved Hamming SEC codes on four groups are represented.

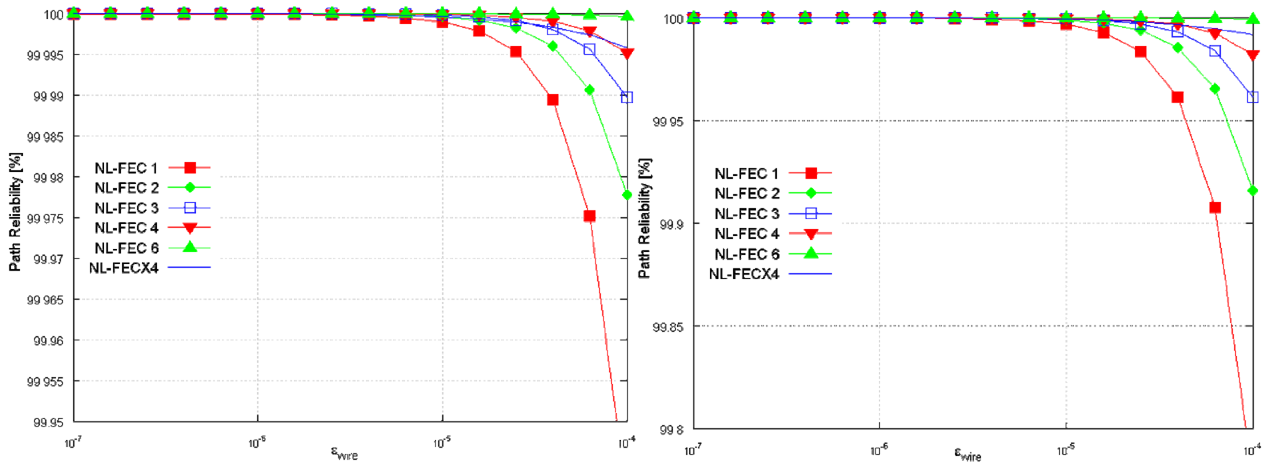


Figure V-15 Comparison of link-level and network-level FEC for 32 bits routers

In both *32*-bits and *64*-bits networks, the path reliability of *NL-FEC* with intermediate correction stages is higher that in the cases where simple *NL-FEC* schemes are used (i.e. correction only at the destination node, *NL-FEC 1*). Although path reliability targets above *99.75%* can be achieved without intermediate correction stages, adding correction stages significantly improves path reliability.

The reliability improvement observed for *NL-FEC* with correction stages is due to the fact that faults are less likely to cumulate over shorter path sections. Hence, the reliability increases when the path sections between correction stages are shorter. For example, reliability targets above 99.999% are achieved when intermediate correction stages are considered every two hops (i.e. *NL-FEC 6*).

NL-FEC with interleaved Hamming SEC codes improves path reliability. In the examples above, the path reliability of *NL-FEC×4* is similar to that of NL-FEC with four intermediate retiming stages (i.e. correction every three hops) for 32-bits. In some cases, SEC interleaving provides better path reliability that SEC with correction stages, as double errors are more likely to cumulate along path sections than multiple uncorrectable errors along the entire path.

5.2.3.2 Area and power overheads

In 3D NoCs with *NL-FEC*, correction stages can be inserted such that faults can cumulate on path sections no longer than p_{MAX}. Hence, up to six ports (i.e. *NORTH*, *SOUTH*, *EAST*, *WEST*, *UP* and *DOWN*) of the seven-port router may have correction stages. In the 65 nm process, a retiming stage is added before each correction stage in order to maintain the *1GHz* network clock frequency. In this case, flit transmission on the *PHY* and error detection / correction are performed in two clock cycles. The insertion of the retiming stage raises the data loss problem: by the time the buffer *FULL* signal of the receiver router arrives at the transmitter router, a flit could be in the correction retiming stage. Therefore, in order to avoid data loss, an extra position is added for input buffers having a correction stage. In TABLE IV, the area and power overheads for the seven-port router with up to six correction stages are summarized.

The area / power overheads increase with the number of error correction stages. The area overheads are due to the larger flit size, the correction logic and the protected input buffers with an extra position. Compared to *NL-FEC*, protecting a single port increases the area overheads by *10%*, while the power overheads almost double. Although the encoding / decoding complexity increases with data size, the overheads of *64*-bits flits are slightly lower than for *32*-bits, as the relative number of error correction bits is *7/64* instead of *6/32*. The correction complexity outweighs the buffer size when the number of correction stage increases. In this case, the area and power overheads exceed *40%*.

TABLE IV AREA / POWER OVERHEADS [%] OF MULTI-LAYER ERROR RESILIENCE

Error correction scheme	32-bits		64-bits	
	Area (%)	Power (%)	Area (%)	Power (%)
SEC with 1 port	27.64	23.19	26.75	17.83
SEC with 2 ports	32.43	29.01	31.83	23.81
SEC with 3 ports	34.54	30.25	37.77	29.57
SEC with 4 ports	40.15	39.79	43.02	35.28
SEC with 5 ports	45.65	46.3	50.86	42.12
SEC with 6 ports	48.02	48.84	53.97	45.55

In 3D NoCs, it is assumed that correction stages are added only for inter-die links and only two ports have error detection and correction stages. Hence, intra-die and inter-die errors will not propagate between layers, as errors are corrected before the incoming flits are stored in the router input FIFO. Compared to the link-level approach (see all links protection in TABLE II), the area overheads of the multi-level solution are *~5%* larger, but they dissipate *~10%* less power. Although network-level solution with interleaving SEC codes can ensure similar reliability levels, the multi-layer solution is more efficient than *NL-FEC* with two interleaved groups for *32*-bits links. However, the complex Hamming SEC encoding/decoding logic and the buffer protection against flit drop makes the multi-layer solution less efficient for 64-bits. Moreover, interleaving SEC codes also requires more TSVs for error check bits transmission.

5.2.3.3 Impact on network latency

In the multi-layer scheme, adding the correction stages has a negative impact on network latency, as one clock cycle is lost every time flit errors are detected and corrected. Because error detection and correction is performed in a single cycle, the network latency is not affected by the decoder at the destination router. Moreover, there are no variations of link latency with the error rates, as error detection and correction is performed in a single cycle.

Let us consider 3D NoCs with intermediate correction stages inserted for all links and for inter-die links, respectively. In Figure V-16, the impact of correction stages on the network latency is represented for different 3D mesh topologies for uniform random traffic.

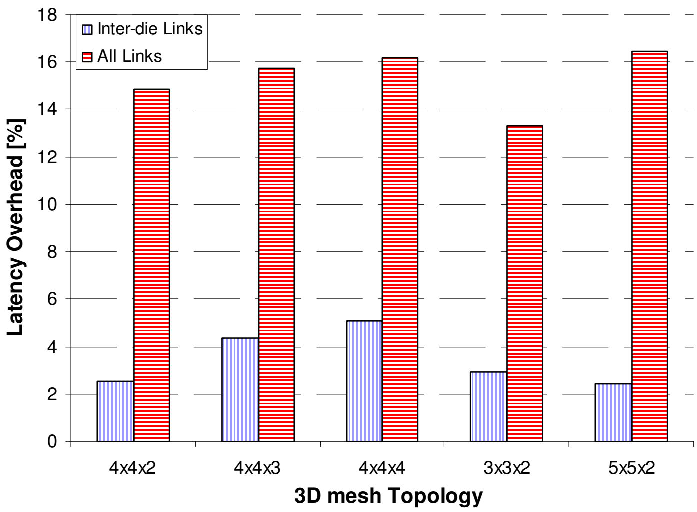


Figure V-16 Impact of intermediate correction stages on network latency

Adding intermediate correction stages increases the link latency by one clock cycle. Hence, the average network latency increases by more than *12%* in the case when all links have correction capabilities. The variations in network latency overheads with the topology are explained by the average path lengths. Flits make on average *2.41* hops for the 3×3×2 topology and *3.78* hops for the 4×4×4 topology. Hence, the performance penalty is higher for network where packets traverse more links with correction stages.

Including error correction only for inter-die links reduces the latency overheads to less than *5%*. Unlike the previous case where all links have correction capabilities, the variations of latency overheads are explained by the average number of inter-die hops, as only paths using such links are penalized. The 5×5×2 topology has relatively high average hop count (i.e. *3.79*), but only half of its paths comprise inter-die links. Therefore, when only inter-die links are protected, the latency overhead is reduced from *16%* to *~2%*. As the number of inter-die links per path increases, so does the latency overhead. In the 4×4×4 topology, *~75%* of paths are inter-die and the latency overhead is *~5%*.

5.2.4 Remarks on mitigating transient faults

In the previous sections, a complete evaluation of error resilient strategies for transient faults was presented. Using the uncorrelated fault model, it was shown how each error control scheme should be configured in order to achieve the 3D NoC reliability target. The results presented in this section are summarized as follow.

At the data link level, error-control schemes improve path reliability by mitigating transients on the physical wires (i.e. intra-die wires and TSVs). For the seven-port router, protecting links using error correction codes (i.e. *Forward Error Correction* schemes) has slightly smaller overheads (i.e. up to 10*%* for all links protection) than retransmission-based schemes that use simpler error detection codes. Unfortunately, using error correction codes requires a significant amount of TSVs for sending the error control bits. Thus, they cannot be used if there are TSV limitations and *ARQ* schemes, which require fewer error check bits, must be used. If error correction is not enough for *FEC* schemes then retransmission capabilities are added. The hybrid error correction and retransmission schemes achieve very high reliability targets, but they incur the higher area / power overheads (i.e. up to *50%* more than *FEC*).

In 3D NoCs it was shown that the overheads can be reduced almost three times by protecting only inter-die links. This area / power benefit also comes with improvements in terms of performance, as the latency delay of inter-die links increases. Despite these benefits, the selective inter-die link protection strategy has limited reliability improvements capabilities, making it suitable for technologies where inter-die links are more likely to be affected by transients than intra-die components (i.e. routers and intra-die links).

Major limitations of link error control schemes include the relatively high performance penalty (i.e. at least one cycle is lost for each hop in the all link protection scheme) and its inability to mitigate transients on router buffers. When adding correction codes for inter-die link is not an issue, these challenges are jointly solved for individual flits using network-level forward error correction (*NL-FEC*). For low error rates (i.e. at most one transient per flit along the path), *NL-FEC* proves more efficient in terms of area and power than all links protection. For higher error rates, interleaving Hamming SEC codes ensure higher reliability with negligible impact on network latency, but the costs increase significantly (i.e. ~20% more area than all links FEC).

An alternative to interleaved codes is a multi-level use of *Forward Error Correction*, where flits are encoded at the source node and error detection and correction is performed at destination and at intermediate correction stages. For high error rates, this solution is better in terms of area / power overheads than *NL-FEC* (i.e. up to 17% smaller overheads). However, using this strategy will have a greater impact on network latency, as error correction takes at least one clock cycle. Hence, when correction is performed for inter-die links, the latency overheads are up to *5%* for 3D mesh topologies.

5.3 *Conclusions*

In this chapter, the error resilience schemes for TSV permanent and transient faults have been analyzed. Experimental results have shown that permanent TSV faults due to manufacturing defects can be efficiently mitigated using link-level and network-level repair. In the cases where no spare TSVs can be used, serialization (*CSL*) and fault-tolerant routing (*TSV-FTR*) provide alternatives to spare-based solutions. Although they ensure high TSV reparability, *CSL* and *TSV-FTR* solutions affect network performance. Adding spares improves network performance, but the costs of serialization with spares (i.e. up to *20%* area overhead) exceed those of fault-tolerant routing with spares. In the multi-layer approach a limited number of spares can be allocated for inter-die links and fault-tolerant routing is more efficient than serialization.

In-field TSV failures are more difficult to repair, as external testers have no access to TSV of the repair fabric. In this case, the Interconnect Built-In Self-Test (*IBIST*) and Built-In Self-Repair and Adaptive Serialization (*IBIRAS*) solutions are implemented on-chip. TSVs are tested during system life-time using KAF-based test patterns and faulty TSVs are repaired using functional spares (i.e. *IBISnR*) or by link serialization. Embedding such capabilities has non-negligible costs, as the router overheads go up to *40%* when the TSV failure rate during system lifetime is $d_{TSV}=1\%$.

Mitigating transients on inter-die links in an efficient way is more difficult, as there are many possible solutions. At link-level it was shown that *Forward Error Correction* is more efficient. This solution has the disadvantage that it requires a significant amount of redundant TSVs for the error control bits. An alternative are retransmission-based schemes (ARQ) that use error detection codes, which requires less error check bits. In this case, the performance penalty increases, as flit retransmissions depend on the flit error rate. When adding error control bits is not an issue, the reliability shortcomings of link-level error control are alleviated by network-level FEC schemes. Flits are individually encoded at source and errors are detected and corrected at destination. Although there is no impact on network performance, as flit encoding and decoding is performed in a single clock cycle, the disadvantage of this scheme is that in unreliable technologies multi faults will cumulate along the path, making error correction impossible or prohibitively expensive. Although SEC code interleaving could solve this issue by having multiple error capabilities, an inter-die multi-layer solution proves more area / power efficient in some cases. Hence, link-level and network-level *FEC* are jointly used to correct errors cumulated along intra-die path sections. This way, higher communication reliability is achieved with less than 5% overhead on network latency.

Despite the work in error resilience schemes, it is not clear which of these solutions is better suited for a given 3D NoC application. In designing an MPSoC, *good practices* may lead to solutions that are too conservative and that lead to over-designed systems. Identifying the best error resilient NoC configuration at design time is a challenging task, as there are many aspects that must be taken into account. An exhaustive analysis of 3D NoC error resilient schemes similar to the one presented in this chapter is time-consuming. Therefore, a strategy that finds the best error resilience configuration is necessary. In the following chapter the *error resilience exploration* process is introduced, along with an implementation for 3D NoCs. Using a highly configurable library of single-/multi-layer fault-tolerant solutions, error resilience is automatically implemented and evaluated for the given 3D NoC architecture. The result of the exploration process is a list of potential solutions, which satisfy the reliability / yield requirements for the targeted fault rates. Using the costs (i.e. area, power, and timing) and performance evaluations, the system designer decides which of these solutions is better suited for the 3D NoC-based design.

Chapter Six

3D NOC ERROR RESILIENCE EXPLORATION

There are many error-resilience schemes for 3D NoCs, but it is very difficult to select an optimal solution. One-fits-all solutions do not exist and an exhaustive search of all error resilience solutions often proves inefficient, yielding strategies that are too conservative. In this chapter, this challenge is addressed by an Error Resilience eXploration (ERX) methodology. Different 3D NoC error resilience strategies are implemented and assessed, with respect to hardware costs (i.e. area, power) and impact on network latency, such that reliability and yield targets are achieved with a limited number of TSVs. This process is performed early in the design phases, helping system designers to choose the best solution that satisfies their requirements. In order to validate ERX and assess the impact of error resilience on system performance, a system-level case study is considered. For a 64-tiles 3D MPSoC, which is interconnected by a 4×4×4 3D mesh NoC, different error resilient configurations are determined using ERX. The best solution is chosen and the system-level impact of error resilience is assessed using a SystemC simulation platform based on cycle-accurate SoCLib library.

6.1 *Error resilience exploration for 3D NoCs*

Designing complex 3D systems with reliability and yield targets becomes increasingly difficult when the number of cores per chip goes in the order of hundreds or even thousands. In literature, there are many schemes that allow system designers to mitigate different faults. 3D NoCs can be designed to achieve different yield / reliability targets using one or more of the existing fault tolerant solutions presented in Chapter 4. Unfortunately, designing systems to be reliable often leads to non-negligible hardware costs and performance penalties that are difficult to accurately assess. In the early design stages, finding an optimal solution is very difficult. The alternative is to perform an *error resilience exploration* process, where different fault tolerant strategies are assessed early in the design stage such that designers may chose an optimal solution. In this section, the process of exploration is detailed for 3D NoCs.

In Chapter 2 several NoC design strategies have been briefly presented. In both academic and industrial environments, NoCs are often designed using dedicated tools that automatize many steps of the design process. For example, the design flow of Spidergon STNoC [DCC11] allows designers to build a NoC-based system with different topologies and then generate the simulation platform for the targeted system. Although such tools enable architecture exploration, they give very little information on how reliability could be achieved[1]. Using a similar approach it is possible to generate 3D NoC error resilient configurations that satisfy different yield / reliability targets. These configurations can then be assessed using existing design flows such that the system designer can choose one of these strategies by comparing different performance-costs results.

A 3D NoC design flow has been built around the fully synchronous seven-port router (see Appendix). After the designer chooses the size of the 3D mesh (i.e. the X, Y, and Z dimensions) and the data size, the RTL code of the NoC is generated. For this 3D mesh NoC design flow, an error resilience exploration tool (*ERX*) has been developed. In the following chapter, the architecture and functionality of *ERX* are described.

6.2 *The ERX tool*

The error resilience exploration (*ERX*) tool for 3D mesh topologies finds different configurations of the single-/multi-layer strategies presented in Chapter 4, which are implemented to the original NoC such that reliability and yield targets are achieved. In Figure VI-1, the diagram of the 3D NoC error-resilience exploration flow is presented.

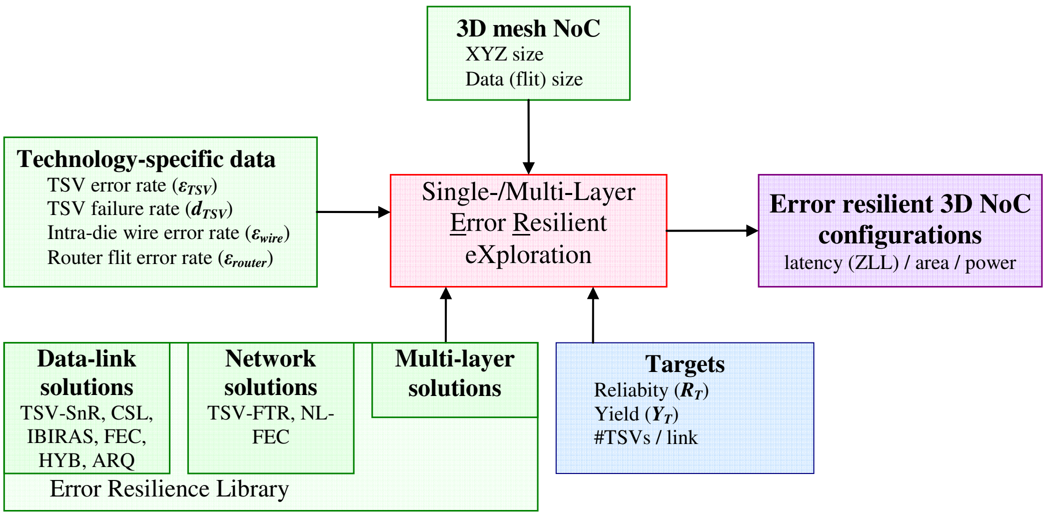


Figure VI-1 Error-resilience exploration (ERX) diagram for 3D NoCs

In *ERX*, a 3D mesh NoC is defined by its size (i.e. number of nodes in the X,Y, and Z directions) and the number of data bits. In order to configure different error resilience schemes, the TSV failure rate (i.e. probability of permanent faults due to manufacturing and aging defects) d_{TSV}, the TSV error rate ε_{TSV} (i.e. the probability of a transient on a single TSV), the intra-die wire error rate ε_{wire}, and the flit error rate / router ε_{router} are given as technology-specific inputs. Each error protection strategy is configured such that the reliability and yield targets R_T and Y_T are achieved. Moreover, it is possible to select configurations that use up to a certain number of TSVs / link.

After the exploration process, the tool returns a list of potential solutions with the impact on the network average latency. The network latency is a theoretical estimation, which is determined using the zero-load

[1] In the case of Spidergon STNoC, a possible error resilient extension to the synchronous Spidergon STNoC links is presented in the Appendix

latency (*ZLL*) model. Using the list of potential solutions, the RTL code of the error resilient NoC can be generated and further cost / performance evaluations can be performed. In the rest of this section, the selection process of error resilience strategies and the cost / performance assessment methodologies are presented.

6.2.1 Error resilience scheme selection

In the error resilience exploration process, fault tolerant schemes for transient faults are configured first, as data and error check bits are transmitted on regular TSVs. The tool may return one or more solutions for each data link and network level, or even some multi-layer solutions. The network reliability is given by the minimal reliability of all possible paths in the network (i.e. the minimal two-terminal reliability). When homogenous network components (i.e. there is no reliability variation of routers and links with the spatial distribution) are considered, the minimal path reliability is given by the longest path in the NoC. Thus, the network reliability for a single flit can be expressed using the error probability on routers ε_{router}, intra-die links $\varepsilon_{link\text{-}2D}$, and inter-die links $\varepsilon_{link\text{-}3D}$, as shown below:

$$\rho_{flit} = (1-\varepsilon_{router})^{x+y+z-2} \cdot (1-\varepsilon_{link-2D})^{x+y-2} \cdot (1-\varepsilon_{link-3D})^{z-1} \qquad \text{(VI-1)}$$

Data link and network protection schemes are configured such that $\rho_{flit} \geq R_T = 1-\varepsilon_T$. Data link solutions are determined first using the selective inter-die link protection strategies (i.e. minimize $\varepsilon_{link\text{-}3D}$). If the targeted reliability level cannot be achieved for any error control scheme (i.e. forward error correction *FEC*, automatic retransmission query *ARQ*, and hybrid error correction and retransmission *HYB*) then an all-link strategy is considered (i.e. minimize $\varepsilon_{link\text{-}2D}$ and $\varepsilon_{link\text{-}3D}$). The reliability of *FEC / ARQ / HYB* protected intra-die and inter-die links are given in Chapter 5, equations (V-7, V-10, and V-12).

Inter-die link *FEC* configurations that satisfy the reliability requirements are selected if there are enough spare TSVs for the error check bits. If the reliability improvements of *FEC* are not high enough then retransmission is added by implementing *HYB* schemes. Since *FEC/HYB* schemes use Hamming SEC/SECDED codes with *one*, *two*, or *four* interleaved groups, a significant amount of wires is necessary for sending these error control bits. When the number of TSVs per link is limited, these solutions cannot be implemented. The only alternative being *ARQ* protection, which uses error detection codes (i.e. interleaved parity codes) with fewer error check bits (i.e. at least two parity bits).

In the case of link protection, transients that affect flits during router traversal propagate along the path, causing serious reliability issues. If there are enough TSVs to accommodate the error check bits of each flit then network-level error correction (*NL-FEC*) can be used. In this case, the interleaved Hamming codes may have *one*, *two*, or *four* groups, depending on the error rates and reliability targets. In the multi-layer scheme error correction stages are added for inter-die links.

At the end of the reliability assessment process, a list of potential solution is returned. For each solution, one or more TSV repair strategies are determined. In Figure VI-2, the selection process for the repair schemes is presented.

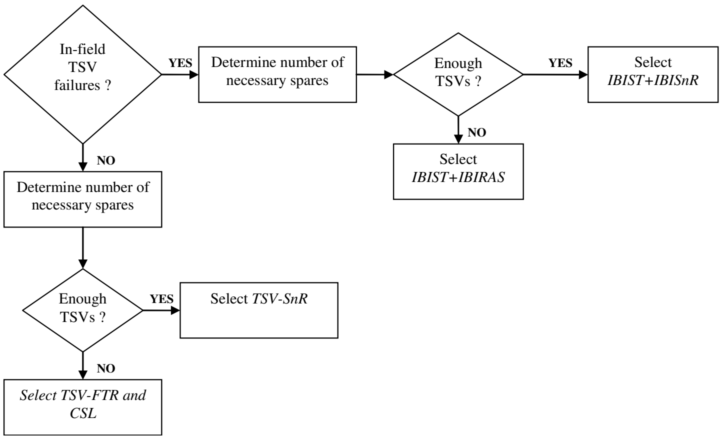


Figure VI-2 Selection process for TSV permanent faults

If only manufacturing faults are considered then spare-based (*TSV-SnR*), serialization-based (*CSL*) and fault tolerant routing (*TSV-FTR*) solutions can be considered. In the case of *TSV-SnR* repair, the number of spares / link necessary to achieve the reliability target is determined. If the number of spares is less than the maximum number of TSVs / link then more spares are allocated and the grouping strategy is used for cost reduction. For example, if at least three spares are necessary to achieve the yield target, but there can be up to four spares then the regular TSVs are split in two / four groups with two / one spare per group. Then, the configuration with fewer spares / group that satisfies the yield target is selected. If the number of spares exceeds the TSV count limit then *CSL* and *TSV-FTR* are used with as many spares as possible. When TSV failures during system life-time are considered, *IBIRAS+IBIST* schemes are used if there are not enough TSVs for spare-based repair (i.e. *IBIST+IBISnR*).

For each error resilient configuration, the impact on network latency is analytically determined for the XYZ mesh NoC using the *Zero Load Latency* model. *ERX* can be used with existing ASIC flows for area and power estimations. In the following section the *ZLL* performance estimation methodology of network latency is presented.

6.2.2 ERX network average latency analytic evaluation

ERX determines the average latency for each protections scheme using the *zero load latency* (ZLL) model. This model allows fast estimation of the impact of error resilience on the network latency. For a packet that traverses the network along a path with p hops, the latency L_{path} is defined as the sum of the latency on all links and routers along the path (i.e. $p\cdot\delta_{link}+(p+1)\cdot\delta_{router}$) and the packet serialization delay δ_{ser} (i.e. the delay necessary to inject the entire packet *flit-by-flit*). Using these notations, the packet latency is expressed as:

$$L_{packet} = (p+1)\cdot\delta_{router} + p\cdot\delta_{link} + \delta_{ser} \qquad \text{(VI-2)}$$

The router delay δ_{router} represents the time necessary for the flit to traverse the router. When links are not protected, it is considered that one clock cycle is necessary for flits to traverse it (i.e. $\delta_{link}=1$). Because encoding and decoding within one clock cycle is not possible, the latency of protected links is at least two clock cycles. Assuming that Hamming detection and correction can be done in a clock cycle, the latency of *FEC*-protected links is two cycles (i.e. $\delta'_{link}=2$). In the case of *ARQ*-protected links, error detection enables the retransmission mechanisms. When errors are detected, the link latency increases to four cycles (i.e. $\delta'_{link}=4$).

Flit retransmission is used in HYB scheme only when the error cannot be corrected. In this case, the link latency increases to five cycles (i.e. δ'_{link}=5), as a retiming stage must be included between the decode output and the retransmission logic input. In general, when ρ_{link} represents the fault probability of the encoded flit, the link latency is expressed as:

$$\delta'_{link} = \rho_{link} \cdot \delta_{link}^{fault-free} + (1-\rho_{link}) \cdot \delta_{link}^{fault} \quad \text{(VI-3)}$$

When all links are protected, the path latency increases by at least p cycles. In the selective link protection strategy, protecting fewer links along the path reduces the overall path latency. For n_p out of n-1 protected links, the path latency of a single flit with selective link-level protection is expressed as:

$$L_{flit} = (p+1) \cdot \delta_{router} + n_p \cdot \delta'_{link} + (p-n_p) \cdot \delta_{link} \quad \text{(VI-4)}$$

Network-level protection is aimed at protecting flits along the source-destination path. In network-level FEC, Hamming SEC encoders are added (i.e. δ_{enc}) on the router local input ports and decoder (i.e. δ_{dec}) on the local output ports. Hence, the path latency is expressed as:

$$L_{packet} = (p+1) \cdot \delta_{router} + p \cdot \delta_{link} + \delta_{ser} + \delta_{enc} + \delta_{dec} \quad \text{(VI-5)}$$

In most cases encoding and decoding can be done within one clock cycle and it is assumed that one cycle is needed for encoding (δ_{enc}=1) and one another cycle is needed for decoding (δ_{dec}=1). However, implementations where there is no penalty on network latency are possible.

In the multi-layer scheme, error correction stages are inserted, in order to avoid error propagation and accumulation along the source-destination path. In the general case, when there are c intermediate retiming stages, the path latency increases by c decoding delays.

$$L_{packet} = (p+1) \cdot \delta_{router} + p \cdot \delta_{link} + \delta_{ser} + \delta_{enc} + (c+1) \cdot \delta_{dec} \quad \text{(VI-6)}$$

Permanent interconnect faults are mitigated at link-level using spares and serialization. While spare-based repair has negligible impact on link latency, serialization increases link latency. For the p-hops path, let us consider that there are s links that serialize data transmission in $T_1, \ldots, T_s$ cycles. Hence, the header flit latency is given by:

$$L_{packet}^{header} = (p+1) \cdot \delta_{router} + (p-s) \cdot \delta_{link} + \sum_{i=1}^{s} T_i \cdot \delta_{link} \quad \text{(VI-7)}$$

When flits are serialized by a *CSL* (or *IBIRAS*) link, they no longer form a continuous stream when they leave the link. This means that even if data is initially sent as a continuous stream (i.e. one flit / cycle), they do not arrive in a continuous stream. In the case of two-cycle transmission, the first data block arrives with a delay of one clock cycle and the subsequent flits leave the link every two cycles. Similarly, when data is serialized in four cycles, the delay of the first flit is three cycles and the remaining flits arrive every four cycles.

This behavior will affect the packet latency, as flits will arrive at destination every T_{max}-1 cycles, where T_{max} is the maximal serialization rate of all links along the path (i.e. $T_1, \ldots, T_s$). For example, when a link with T=2 is followed by a link with T=4, flits arrive at the second link every two cycles and create backpressure, filling the router buffers. Thus, flits will exit the T=4 serializing link every four cycles. In the other case, when a *T=4* link is followed by a *T=2* link, flits arriving every four cycles are delayed by two extra cycles. Since serial transmission ends before the new flit arrives, the relative distance between consecutive flits remains the same. Assuming that the serialization delay is one cycle per flit, the latency of a packet with *F* flits (i.e. $\delta_{ser}=F$) is given by:

$$L_{packet} = n \cdot \delta_{router} + (n-1-s) \cdot \delta_{link} + \sum_{i=1}^{s} T_i \cdot \delta_{link} + T_{\max} \cdot F \qquad \text{(VI-8)}$$

Fault-tolerant routing is another solution to the TSV permanent faults problem. In this case, alternative paths, which contain only functional components, are determined between source and destination nodes. These alternative paths are not always minimal and *d* extra hops through intermediate nodes are performed. In this case, the packet latency is given by:

$$L_{path} = (n+d) \cdot \delta_{router} + (n+d-1) \cdot \delta_{link} + \delta_{ser} \qquad \text{(VI-9)}$$

These theoretical results are used by *ERX* to estimate the impact of the selected error resilient configurations on the average network latency (i.e. the average of all paths in the network). However, such analytical results may not be accurate enough. For more accurate performance assessments, *ERX* can forward the instantiated RTL model of the error resilient NoC to the simulation platform (see Appendix). In order to make the 3D NoC instantiation processes possible, the error resilience strategies used by *ERX* have been implemented as generic and highly configurable RTL modules (see Appendix). In the following section, the assessment methodology of the 3D router area and power is presented.

6.2.3 ERX area evaluation

The *ERX* tool estimates the costs of the error resilience strategy for the seven-port router used in the 3D mesh NoC design flow. In order to assess the costs of error resilience, the RTL model of the fault-tolerant seven-port router with error resilient interfaces is synthesized using the technology library provided by the designer.

In the current version of the tool, the area /power evaluations are performed using an ASIC design flow based on Synopsys Design Compiler®. Since no libraries for TSV placement are used, the area estimations for the TSV footprint are determined from the pitch p_{TSV} (i.e. area of a single TSV is p_{TSV}^2). In Figure VI-3, the evaluation process is presented for the schemes determined by *ERX*.

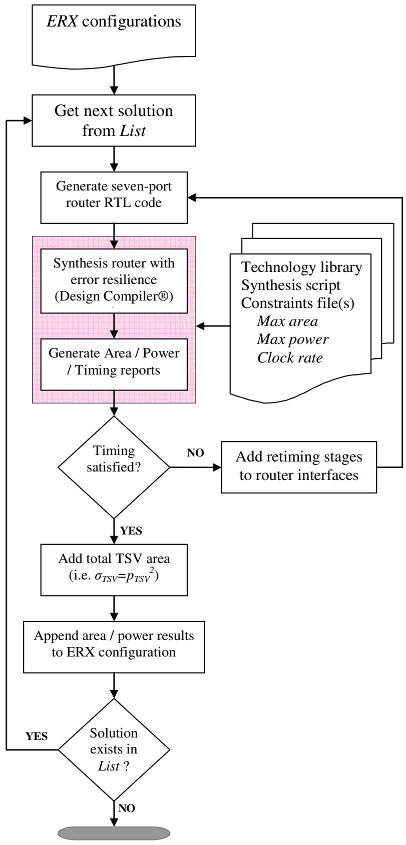


Figure VI-3 Error resilience 3D NoC evaluation flow

The hardware assessment process takes the list of *ERX* solutions and generates the RTL code for the modified seven-port router using the original router and the highly configurable library of error resilience schemes (see Appendix). This library comprises the implementations of the error resilience strategies presented in Chapter 4. Due to its high configurability, the number of retiming stages in each error resilient interface may vary. These retiming stages are inserted in order to improve circuit timing. For example, it is possible to have configurations where SEC/SECDED detection and correction are performed in a single cycle, or they are performed in separate cycles with an optional correction bypass stage.

The area and power estimations are performed using interface configurations with the lowest number of retiming stages. If this configuration does not satisfy the timing constraints imposed by the designer then retiming stages are gradually introduced until no timing violations are detected. For this configuration, the area (i.e. area estimated by Design Compiler® and the TSV footprint) and power estimations are appended to the results returned by the tool.

At the end of the assessment process the system designer has a series of solutions that satisfy requirements in terms of reliability / yield, timing and number of TSVs. Using these results he may choose a solution with minimal impact on network timing, or minimal router area / power overheads. To illustrate the use of *ERX* tool, a system-level assessment is presented in following section.

6.3 *System-level evaluation*

One of the most promising applications of 3D integration is logic-on-logic systems for massively-parallel MPSoC (i.e. SoC comprising hundreds or even thousands of processing elements – MP2SoC). Developing such MPSoCs is outside the scope of this thesis. However, there are open-source simulation platforms that enable system designers to experiment novel concepts like MP2SoC, parallel programming, etc. One of these platforms is the SoCLib library [SCL], which comprises a series of SystemC cycle-accurate bit-accurate (CABA) models for microprocessors (e.g. ARM, MIPS, SPARC), memory controllers, interconnect fabrics (e.g. PI-bus, DSPIN, crossbars), and other peripherals (e.g. terminal emulators, interrupt controllers, DMA, timers, etc). In this section, the capabilities of *ERX* are proven for a 3D shared-memory system modeled using SoCLib. The 3D MPSoC consists of clusters arranged on a regular 3D mesh, which are globally interconnected by a 3D mesh NoC. The *ERX* solutions are determined for different scenarios. One of these solutions is chosen by the system designer and its impact on system performance is assessed.

6.3.1 3D System Architecture

For MP2SoC applications, the SoCLib library proposes a cluster/tile-based approach, where local / global communication is performed using the VCI [VCI01] protocol. In the VCI standard, communication is performed using load / store transactions between an initiator (e.g. CPU, DMA) and a target (e.g. memory controller). The cluster-based architecture has two levels of interconnect fabric: the local fabric, which is interconnecting modules within a cluster, and the global fabric, which ensures inter-cluster communication.

The shared-memory 3D MPSoC consists of clusters interconnected by a 3D mesh NoC. Within each cluster, IP blocks communicate using the local crossbar. Each cluster contains at least one master (e.g. CPU) and one slave (e.g. memory controller, timer, TTY terminal, memory locks, etc.). Inter-cluster communication is performed through the 3D NoC whose network interface controller (NIC) is connected to the local crossbar and translates local transactions to NoC packets. The clusters are planar structures (i.e. they are implemented using 2D technologies) and only the 3D NoC spans across the silicon stack. In Figure VI-4, a two-layer 3D MPSoC with two clusters in each layer is represented.

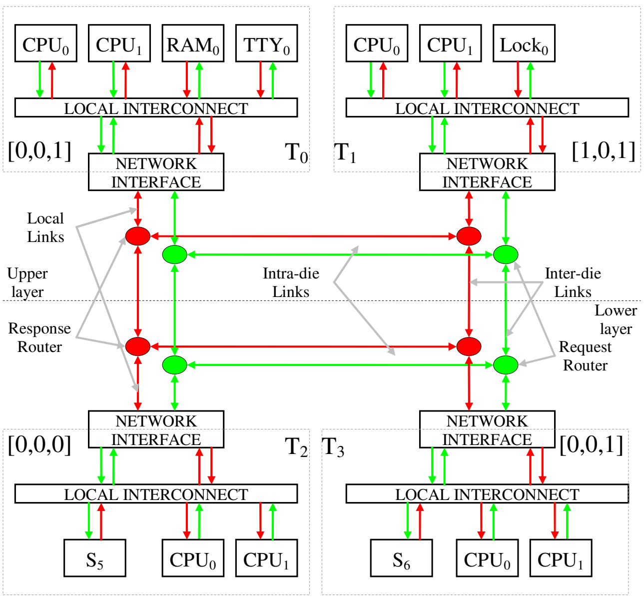


Figure VI-4 3D MPSoC architecture with physically separated request and response networks

The *NIC* of each cluster has both initiator and slave interfaces that are connected to the request and response networks. Physical separation of request and response networks is necessary in order to avoid high-level (i.e. transaction-level) deadlocks. In cluster T_0, load / store transaction between CPU_0 and RAM_0 are carried on the local interconnect. If CPU_0 from cluster T_0 initiates a transaction with the slave device S_6 from cluster T_3, then transaction messages go through the network interface *NIC* of T_0. These messages are translated into packets that are processed by the request network. The packetized request arrives at the network interface controller of T_3 and then it is forwarded to slave S_6, which processes the request. The response returned by S_6 is packetized by the local *NIC* and it is sent through the 3D NoC to the initiator CPU_0 in T_0.

Virtual systems based on the SoCLib library are capable or running a series of applications, including full operating systems like eCos, RTEMS, MutekH, and DNA-OS. Unfortunately, the cycle-accurate simulation execution times increase exponentially with the number of CPUs, making such simulation impractical. Thus, simple applications that access different memory locations (i.e. global and local memory) are considered for system-level performance assessments.

In the 3D mesh cluster configuration, each cluster is identified by a unique set of *X,Y,Z* coordinates. Within each cluster there are two processors and different slave devices such as on-chip memory, timers, memory locks, etc. Access to off-die memory is done through the memory controller RAM_0 in cluster *(0,0,0)*. In order to maintain compatibility with the 32-bits components of the SoCLib library, the request network flit size is 37 bits (i.e. 32 data bits, four byte enable and one side-band information bit for *End-of-Packet*) and the response network size is 33 bits (i.e. 32 data bits and one *EoP* side-band information bit).

Since *SoCLib* does not support 3D architectures, 3D mesh network interface controller (NIC) and fully-synchronous seven-port router SystemC models have been implemented. The system designer can use these extensions to build 3D MP2SoC with different mesh topologies.

For 3D systems comprising hundreds or thousands of cores, reliability and yield become serious challenges. Assuming fault-free intra-die and inter-die communication is no longer possible and error resilience strategies must be used in order to achieve the desired reliability / yield targets. The system designer may use the *ERX* tool to identify potential error resilience solutions for the 3D mesh request and response network. In the following section, several case studies are presented for an MPSoC with 64 clusters interconnected by a $4\times4\times4$ 3D mesh NoC.

6.3.2 Error resilient configurations

In the system-level evaluations, an MPSoC with 64 clusters is considered. Each of these clusters has up to four CPUs, a small on-chip memory and other components such as timers, interrupt / terminal controllers, and DMA. The $4\times4\times4$ 3D NoC global interconnect fabric consists of two physically separated networks with *33*-bits and *37*-bits flits. The *ERX* tool is used to identify error resilient configurations for the request and response networks. The objective is to achieve the *YT=99.99%* TSV reparability (yield) target and the *RT=99.9999%* reliability target (i.e. $\varepsilon T=1e\text{-}6$).

The experimental results of ERX are presented in the following for two experimental setups. Synopsys Design Compiler® was used for area / power / timing estimations for a *65 nm* low-power technology. The synthesis process of the error resilient router was performed for a target clock rate of *1GHz*.

6.3.2.1 *First 3D MPSoC configuration*

In the first experimental setup, let us consider that errors on routers are negligible (i.e. $\varepsilon router=0$), the error rates for intra-die and inter-die wires are $\varepsilon_{wire}=10^{-9}$ and $\varepsilon_{TSV}=10^{-6}$, respectively. In this setup, the TSV failure rate is $d_{TSV}=0.01\%$, no TSV in-field failures are considered, and up to *10* TSVs may be added for each inter-die link. Note that *4* of these spare TSVs are used for TMR protection of flow control signals *write* and *write_ok*, leaving only *6* TSVs for data signals.

In the non-protected case, the probability that a single flit correctly arrives at destination is $\rho_{flit}=99.988\%$ for the request network, $\rho_{flit}=99.989\%$ for the response network. This minor difference is due to the different flit size: in general, the error rate of a flit with *n* bits is $\varepsilon_{flit}=1\text{-}(1\text{-}\varepsilon_{wire})^n$. In the uncorrelated fault model, it is more likely for larger flits to be affected by errors. In the initial configuration, the TSV yield (i.e. probability that all TSVs of all inter-die links are functional) is $Y_{3D}=49\%$ Using *ERX*, the yield / reliability targets are achieved by the configurations presented below.

In the first configuration, protection against transients on data bits is provided only on inter-die links using *ARQ* with interleaved parity on four groups (i.e. *4* TSVs for error check bits). Permanent TSV faults due to manufacturing defects are repaired using two spare TSVs. Hence, the regular TSVs of each link are split in two groups and a single spare is allocated for each group. The area / power overhead of this configuration are *14%/10%* for the request network routers, and *12%/9.5%* for the response network routers. The network latency increases due to transients on inter-die links, as retransmissions are performed every time an error is detected. The estimated latency overhead is *2.5%* for the request network and *2.49%* for the response network.

In the second configuration, inter-die links are protected using *FEC* schemes with Hamming SEC codes. In this case, all the remaining *6* spare TSVs are used for error check bits transmission. The TSV manufacturing defects are repaired using the *TSV-FTR* routing algorithm. The area / power overheads of this configuration are *19%/12%* for routers in the request network, and *18.85%/11.55%* for routers in the response network. In terms of performance, the network latency increases for packets traversing inter-die links, as the link latency increases by one clock cycle and the path length increases in the case when packets are routed around faulty links. Hence, the latency overhead is estimated at *2.2%* for the request network and *2.17%* for the response network.

In the third configuration, inter-die links are protected using FEC schemes with Hamming SEC codes, while TSV manufacturing defects are repaired using link-level serialization (i.e. *CSLs*). The area / power overheads of this configuration are *19.5% / 12.3%* for the routers in the request network, and *18.985% / 11.65%* for the routers in the response network. In terms of performance, the network latency increases for packets traversing inter-die links, as the link latency increases by one clock cycle. Moreover, inter-die links with faulty TSVs serialize data in two cycles, increasing the link latency to three cycles. Hence, the latency overhead is *~2%* for the request network and *2.1%* for the response network.

The configurations presented above have a negligible impact on network latency. Thus, the decision to implement one of these strategies is based only on the area / power overheads. Due to its lower overheads, the first configuration is the obvious choice.

6.3.2.2 Second 3D MPSoC configuration

In the second setup, let us consider that flits are affected by errors with the probability $\varepsilon_{router}=10^{-8}$, the error rates on intra-/inter-die wires are $\varepsilon_{wire}=\varepsilon_{TSV}=10^{-6}$, and the in-field TSV failure rate is $d_{TSV}=0.1\%$. In this setup, the probability that flits arrive at destination fault-free is $\rho_{flit}=99.96\%$ for the request network and $\rho_{flit}=99.97\%$ for the response network, while the inter-die link reparability (i.e. the probability that all links are functional during system lifetime) is less than *1%* (i.e. $0.9999^{96 \cdot 37}$). The error resilient configurations returned by the *ERX* tool are summarized below.

In the first configuration intra-die links are protected using *forward error correction* (FEC) with Hamming SEC codes, while inter-die links are protected using retransmission-based schemes with error detection codes (i.e. *ARQ* with two parity bits). Because there are some redundant TSVs available, permanent faults may be repaired using spares (i.e. *IBISnR*). Hence, two spares are allocated for each inter-die link of the 3D NoC. The area / power overheads of this configuration are *40%/35%* for the request router and *41%/36%* for the response router. The network latency increases, as encoding and decoding modules are inserted on all links. The estimated latency overhead is *~8%* for the request network and *7.95%* for the response network.

In the second configuration, all links are protected using *FEC* with Hamming SEC codes. However, as most redundant TSVs are used for sending the error check bits, TSV permanent faults are repaired using serialization (i.e. *IBIRAS*). The seven-port router area / power overheads for this configuration are *42% / 37%* for the request routers, and *42.5% / 37.7%* for the response routers. The network latency estimations for this case are estimated to *10%* for the request network and *9.85%* for the response network.

In the third configuration, network-level FEC (*NL-FEC*) protection with Hamming SEC codes is used for transient faults mitigation. Since there are no available TSVs for spare-based repair, permanent faults are repaired using serialization (i.e. *IBIRAS*). The area / power overheads are *43% / 35%* for the request network and *43.5% / 36%* for the response network. The *ERX* tools estimates an average latency overhead of *3%* for the request network and *2.9%* for the response network.

The on-chip TSV test and repair capabilities increase the router area and power overheads above *40%* and *30%*, respectively. Although the proposed error resilient configurations have similar overheads, the third configuration is more likely to be chosen, since its impact on network latency is very small.

In this section, two 3D MPSoC case-studies have been considered for the ERX tool. Optimal solutions, with respect to area / power overheads and estimated network latency, have been identified. In order to assess the system-level impact of error resilience, these configurations have been implemented in SoCLib for system-level evaluations. These results are presented in the following section.

6.3.3 Performance evaluations

The system performance is affected by the error resilience schemes, as the latency of packets traversing the network increases. In the previous section, the impact of error resilience on network latency was determined using analytical solution based on the zero-load latency model. In this section, the accuracy of these results is enhanced by analyzing the impact of error resilience on the full system. In order to assess the system-level impact on system performance, the *optimal* ERX solutions determined in the previous section have been implemented in the SoCLib library.

Although the 3D platform modeled using SoCLib is able to run complex software applications (including OSes), the simulation times are very long for the 64-clusters MP2SoC. Therefore, in order to mimic the system functionality, each processing element executes a series of read/write memory operations. This way, both the local and the global interconnect fabrics are loaded with transaction messages.

In the first error resilient setup, errors are injected only for inter-die links (i.e. $\varepsilon_{TSV}=10^{-6}$), since the error rate for intra-die links is very small and no intra-die link protection strategy is considered. Thus, flits on inter-die request / response links are retransmitted with a *~0.35%* probability (i.e. the probability that errors are detected on at least one link). Experimental results have shown that the performance (i.e. application execution time) penalty is negligible (i.e. less than *0.1%*).

In the second setup, the system comprises *9600* TSVs and up to *10* of these TSVs could fail during system lifetime. If these faults affect the TSV used for flit transmission then the affected inter-die links are configured to serialize data in two cycles. In the worst case, these faults affect *10* different links, which will serialize data and cause maximal performance penalty. Experimental results have shown that the performance (i.e. application execution time) penalty is higher than in the previous case (i.e. *~1%*).

The main contributor to the system performance penalty is due to the longer load/store protocol execution times. In Figure VI-5, the execution of a single transaction between an initiator and a target is represented for the initial (i.e. unprotected) and error-resilient setups.

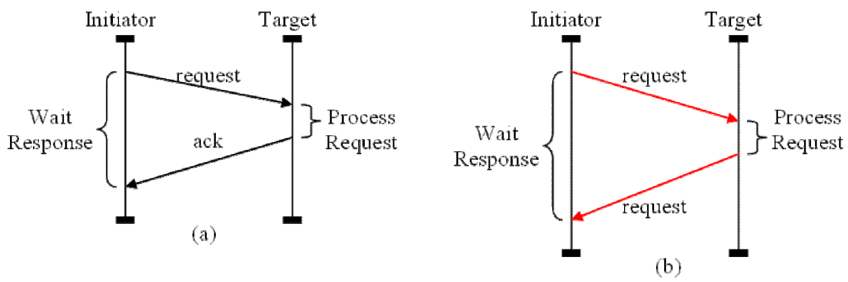


Figure VI-5 Transaction execution time in the (a) unprotected and (b) error-resilience platform

In the error-resilient configuration, the extra delay of the protocol execution time is determined by the longer times required by the request and response packets to the networks. The small performance penalties (i.e. less than *1%*) are explained by the fact that the 3D network traversal takes less than *10%* of the transaction execution times.

6.4 *Limitations of error resilience exploration*

Error resilience exploration is an early-design generic methodology whose limitations are briefly discussed in this section. First, it cannot be implemented into an existing NoCs design flow in an implementation-agnostic way. In order to support the *ERX* instantiation and assessment strategies, the 3D NoC router and (error resilient) interfaces must be carefully implemented. The 3D NoC design flow used in this chapter has been extended with highly configurable error resilient interfaces (see Appendix). In general, adding such capabilities to the NoC design flow do not require any major modifications. An error-resilience framework for

Spidergon STNoC links is presented in the Appendix. This framework is then used in a full-system assessment using customized MPSoC designs from the Gaisler IP library.

Being an early-design tool, *ERX* cannot accurately predict the area / power dissipated by the network. The area used by *ERX* comprises the circuit area estimations given by the synthesis tool and the TSV footprint determined using the TSV pitch. The power estimations are also given by the synthesis tool for a *100%* switching activity on logic gates. Depending on the CMOS library, these estimations do not take into account the power dissipated on wires. Modifications to the area / power estimation methodologies are possible. For example, more accurate RTL-level and gate-level power estimations can be obtained using different netlist back-annotation and standard activity files.

Another limitation of the *ERX* implementation is that it works only for 3D mesh topologies that use the fully-synchronous seven-port router architecture (i.e. no virtual channels, dimension-order routing, X_{ON} / X_{OFF} flow control). Also, *ERX* considers only inter-die wire failures, as pre-bond testing strategies were assumed. Thus, alternative error resilience solutions based on fault-tolerant routing, where intra-/inter-die links and routers may fail, are not explored.

6.5 *Conclusion*

Error resilience exploration for 3D NoCs is an early-design process that helps designers achieve reliability and yield targets in NoC-based MPSoCs. This process consists in quick estimations of the hardware costs and performance penalties for 3D NoCs with different error resilient configurations. In this chapter, an *ERX* methodology and tool for 3D mesh NoCs has been presented.

In order to show the *ERX* capabilities and assess the system-level impact of error resilience, *ERX* has been jointly used with an MPSoC design flow, which uses the SystemC SoCLib library. Using a 64-tiles 3D MPSoC partitioned on four stacked dies, the *ERX* tool found several configurations that satisfy the reliability / yield requirements of two experimental setups. By comparing the costs and impact on network performance of the configurations returned by *ERX*, a *best* solution was chosen. This solution was then *manually* implemented using the SoCLib library and a system-level simulation was performed in order to assess the impact of the selected error resilience strategy on system performance. The results have shown that despite the significant router area / power overheads, having error resilience does not have a major impact on system performance.

Overall, it can be concluded that, for systems with reliability requirements, *ERX* enables designers to quickly identify the best solution. The costs of data link level and network level error resilience can be minimized by careful exploration, while the impact on system performance can be negligible.

Chapter Seven

CONCLUSION AND FUTURE WORK

In very-deep sub-micron (VDSM) technologies, the power and delay problems of global wires can be alleviated at technological level by stacked 3D integration. From the system design perspective, 3D MPSoCs integrate more capabilities in a single chip. Interconnecting an increasing number of IP blocks, while ensuring high computation and communication throughput, requires a scalable solution: the *Network-on-Chip*.

Despite the recent progress of Through-Silicon-Via (TSV)-based 3D integration technologies, they are not considered mature enough for large-scale production. Testing, manufacturing costs, yield and reliability, thermal management, and heat removal remain major challenges. In 3D NoC-based MPSoCs, intra-/inter-die communication reliability is paramount, as transmission faults during network traversal may lead to system failures. Therefore, efficient interconnect test strategies and error resilience techniques become mandatory. The TSV test, yield and reliability of 3D NoC-centric MPSoCs have been addressed in this thesis.

Testing inter-die wires becomes increasingly difficult when the number of TSVs / chip is in the thousands or tens of thousands range. The *Interconnect Built-In Self-Test* (*IBIST*) strategy is proposed for testing TSVs of inter-die links in 3D NoCs. Among the advantages of this approach is the capability to sensitize both structural faults like opens and shorts, and parametric faults like delay faults and delay faults due to crosstalk. Using the aggressor-victim scenario, a new fault model has been proposed: the K^{th} *Aggressor Fault (KAF)* model. The aggressors of each victim wire are the TSVs within a distance given by the aggressor order *K*. The major advantages of KAF-based testing are shorter test times (e.g. *16* cycles for 1^{st} aggressor orders and an arbitrary number of TSVs) and simple hardware implementation, which are up to *50%* smaller than existing ones based on marching-ones and MAF-based tests. Moreover, a configurable *KAF*-based *IBIST* implementation was also proposed. Despite the relative high costs (i.e. almost twice the size of simple KAF-based implementations) of configurable *IBIST*, being able to modify the aggressor order used for TSV tests has several advantages. First, tests during system lifetime can be performed for lower aggressor orders, reducing the off-line time of inter-die links. This strategy can also be used for calibrating TSV technologies before full-scale production.

The TSV reliability and yield challenges of 3D NoCs can be addressed by a single-layer or a multi-layer approach. In 3D NoCs, the effort of mitigating permanent and transients faults on TSVs can be shared between the data link and network layers. Spare-based repair and serialization are two techniques used for repairing permanent TSV faults in inter-die NoC links. Different implementations have been proposed in order to address permanent faults due to manufacturing defects (i.e. *TSV Spare-and-Replace*, *Configurable fault-tolerant Serial Links*) and interconnect aging / wear-out (i.e. *Interconnect Built-In Self-Repair and Adaptive Serialization*). Network failures due to TSV manufacturing defects are avoided by implementing TSV fault-tolerant routing (*TSV-FTR*) algorithms. In the multi-layer approach, the data link serialization-/spare-based strategies are jointly used with *TSV-FTR* in order to reduce costs and impact on network latency.

Transient faults, which account for most system failures, on intra-/inter-die links are handled by data link strategies based on data coding, error correction (i.e. *Forward Error Correction*) and retransmission (i.e. Aut*omatic Retransmission Query*), or network-level error correction (*NL-FEC*).

In order to assess different trade-offs, the error resilience strategies have been implemented on the fully synchronous seven-port router of 3D mesh NoC. The experimental results have shown that, in most cases, traditional solutions like spare insertion (*TSV Spare-and-Replace*) and data retransmission (*Automatic Retransmission Query*) are not effective for TSV high failure rates. Error correction codes (ECC)-based solutions for link protection against transients prove to be more efficient in 3D NoCs. Instead of protecting *all* links in the network, it has been shown that protecting only inter-die links significantly reduces hardware costs, while the network latency penalty is reduced.

Protection against permanent faults due to manufacturing defects can be ensured without using spares. In this case, links serialize data using the remaining functional TSVs (i.e. *CSL*) or routers deviate packets on alternative fault-free inter-die links (i.e. *TSV-FTR*). In both cases, the yield gain comes at the expense of additional circuitry for serialization or fault-tolerant routing. Experimental results have shown that, for highly defective TSVs, serialization with no spares provides the best performance-cost trade-off, while fault-tolerant routing with spares has lower hardware costs, but slightly higher performance penalty.

TSV permanent faults during system life-time are a serious issue for 3D MPSoC comprising many TSVs with high defect densities. The solution adopted in this thesis is an off-line self-test and repair process at the link level. *KAF*-based Interconnect *BIST* strategies are jointly used with the *Interconnect Built-In Self-Repair and Adaptive Serialization (IBIRAS)* scheme, which ensures both spare- and serialization-based repair. Experimental results have shown that in current technologies serialization-based repair pays-off only when it is not possible to insert the necessary number of TSVs/ link. However, in future technologies the impact of the TSV footprint will be larger than that of the serialization / de-serialization circuitry, making serialization-based solutions more attractive.

Although there are many solutions to the TSV reliability / yield challenges, it is difficult to choose the best strategy for a given 3D system. *One-fits-all* solutions do not exist and each error resilience scheme has its own advantages and disadvantages in terms of costs and impact on system performance. Therefore, an error resilience exploration (*ERX*) tool has been proposed for 3D NoCs. *ERX* takes as inputs the NoC architecture (i.e. topology, flit size, etc), the reliability / yield requirements, and component failure rates (i.e. TSV failure rates due to manufacturing / aging and *bit error rate*s on inter-die / intra-die links). Using the library of highly configurable error resilience techniques, it automatically implements and assesses different error resilience strategies for the targeted 3D NoC. The result is a set of solutions that satisfy the reliability / yield requirements and a series of constraints such as the number of TSVs / chip. Using these solutions, the system designer decides which solution is better fitted for the system.

In an experimental case-study, the capabilities of *ERX* have been evaluated on a 3D massively-parallel MPSoC (MP2SoC) using a SystemC virtual platform based on the SoCLib library. The 3D MPSoC has 64 clusters distributed across four layers and interconnected by a *4×4×4* 3D NoC. For two experimental setups, *ERX* identified several protection strategies, of which one solution (i.e. the optimal one) was implemented in the virtual platform. In both cases, the increased latency of the request and response networks has a negligible impact on the application execution time (i.e. less than 1%). These results show that, despite their non-negligible costs, careful selection of data link and network solutions may not lead to any significant performance overhead.

The solutions presented in this thesis are far from being complete and future directions can be envisaged. It is always possible to implement alternative fault tolerant solutions at data link and network level. Indeed, spare-based repair, serialization, and data encoding are not novel protection strategies and alternative 3D NoC-specific implementations can always be developed. Within the *ERX* framework, these solutions can be easily integrated by adding them in the library of error resilience strategies.

Further extensions of ERX concern the assessment methodology. Throughout this thesis, costs (i.e. area / power) were evaluated using a standard synthesis flow with a *65 nm* library. In future work, this flow can be extended to full 3D environments where assessments are performed in later design stages (e.g. after TSV / circuit place-and-route). Although *ERX* latency measurements are independent on the traffic patterns, the network latency was measured for random uniform traffic or using the zero-load latency (ZLL) model. However, in order to accurately predict the impact on system performance without relying on full-system simulations, the existing traffic models could be extended to support different pseudo-random distributions, synthetic traffic, or even real traffic traces.

It is also possible to implement error resilience at other abstraction layers. At physical level, wire spacing or circuit-level protection strategies can improve overall system robustness. Transport-level retransmission, transaction-retry or software-based fault tolerance are other potential solutions. Although these solutions are outside the scope of *ERX*, which mainly targets 3D NoCs, it is possible to extend such an exploration platform that assesses costs and performance penalties at different abstraction layers.

Chapter Eight

RÉSUME EN FRANÇAIS

L'intégration 3D est une nouvelle technologie qui permet l'intégration hétérogène de plusieurs composants semi-conducteurs sur la même puce. Des couches de silicium sont empilées est connectées par des fils verticaux Through Silicon Vias (TSVs). Cette technologie permet d'améliorer les performances des systèmes intégrés et de réduire la consommation. Malgré ces avantages, les problèmes de fiabilité, de rendement, de tests et de thermique restent parmi les grands défis de cette technologie. Dans cette thèse, les problèmes de rendement, de fiabilité et de test des interconnections TSVs sont traités dans le contexte des Réseaux-sur-Puce 3D (3D NoCs). Une stratégie d'autotest (Interconnect Built-In Self-test - IBIST) est proposée pour les liens 3D. Les problèmes de fiabilité et de rendement des TSVs sont traités dans un contexte uni/multi couche ou l'effort de masquer les fautes est partagé par les couches data-link et network du NoC. Ce chapitre est le résume des travaux présentées dans les chapitres précédents.

8.1 *Intégration 3D et les Réseaux-sur-Puce*

L'intégration 3D est une technologie émergente qui promet l'intégration hétérogène avec une basse consommation et des performances accrues. Les couches de silicium sont empilés et reliés par des fils *Through-Silicon-Vias* (*TSV)*. Aujourd'hui, l'intégration 3D n'est pas encore à pleine maturité et la conception de nouveaux concepts et stratégies d'essai sont en cours d'élaboration. Alors que de nouvelles techniques sont utilisées pour améliorer le rendement et la fiabilité des puces 3D, plusieurs techniques ont été adaptées à partir de domaines bien établis.

Les noeuds des NoCs 3D sont distribuées dans les couches de silicium et sont connecté par des liens intra-die et inter-die, assurant la communication intra et inter-die entre les composants des systèmes. La topologie des NoC est totalement déterminée par la distribution et la connectivité des IPs du système. Pour un système avec *N* nœuds, les topologies régulières, quasi-régulières et spécifiques sont présentées dans la Figure II-12.

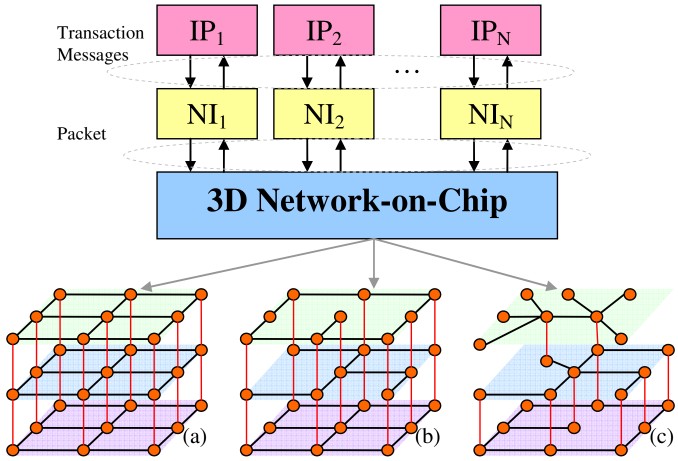


Figure VIII-1 MPSoC implemented using 3D NoCs

La tâche des routeurs est d'envoyer les paquets dans la direction demandée. A chaque routeur, la direction du routage est déterminée après l'inspection de l'entête du paquet. Les routeurs 3D sont des structures planaires (2D) qui ont des interfaces dédiées pour les liens intra-die et inter-die.

Grâce aux liens verticaux, les topologies 3D ont une meilleure connectivité que les NoCs 2D. Plusieurs études ont montré que les performances des NoCs 3D sont nettement meilleures [PF07, FP09, QLD09], surtout pour les topologies *mesh*, *thorus* et *ciliated mesh* [FP09]. Les TSVs sont des ressources très chères est leur coût est non négligeable. Par conséquent, des topologies 3D spécifiques qui réduisent le nombre des TSV utilisées, la puissance consommée et qui améliorent les performances du système ont est été proposées [YL08, SMB09]. Des techniques de sérialisation [Pas09, DVS11] ont été proposées pour réduire le nombre d'interconnections verticales dans les puces 3D.

Le test des interconnections et la tolérance aux fautes des NoCs sont des sujets de recherche très populaires dans la communauté. On trouve beaucoup de stratégies dédiées pour les NoCs 2D qui sont également utilisées dans un environnent 3D. Parmi les objectifs de cette thèse, figurent le test et la fiabilité / rendement de la structure d'interconnexion des systèmes 3D. Des solutions différentes aux problèmes mentionnés ci-dessus sont donc présentées dans les sections suivantes.

8.2*Stratégies de test pour les liens verticales dans les NoCs 3D*

Les liens verticaux des NoCs 3D sont des éléments essentiels, car la plupart des avantages en terme d'efficacité énergétique et de performance sont dues à eux. Dans cette section, les défis de test des liens verticaux sont traités par une stratégie d'autotest: *Interconnect Built-In Self-Test* (*IBIST*). La stratégie de test proposée doit sensibiliser les défauts structurels permanents tel que les *opens* et les shorts, mais aussi de retarder les défauts dus à la diaphonie pendant le fonctionnement du système.

Les connections entre les soutes sont très souvent implémentés par des liens unidirectionnels. Pour un lien vertical entre le routeur émetteur T_X et le routeur récepteur R_X, l'architecture de test *TSV-IBIST* est représentée dans Figure III-2.

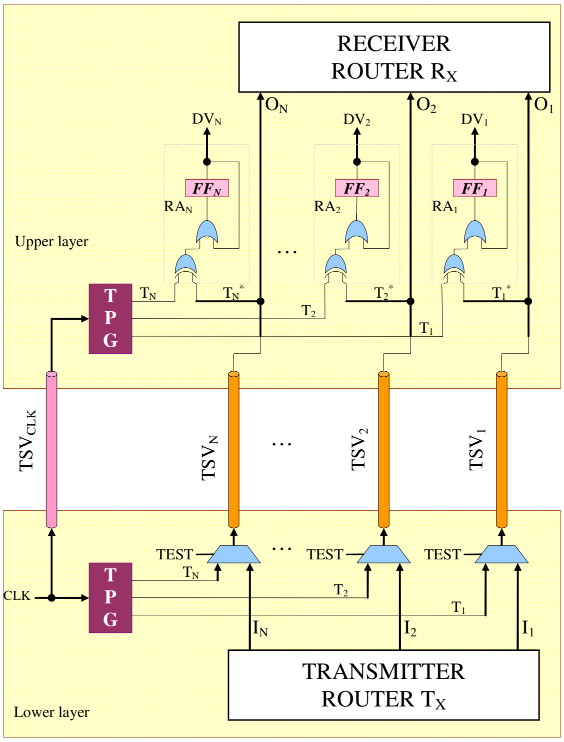


Figure VIII-2 Inter-die Link Interconnect Built-In Self-Test (IBIST)

Pendant le test des interconnections TSV_1-TSV_N (*TEST='1'*), le routeur émetteur est bloqué et les vecteurs de test T_1:T_N sont générés dans la couche en bas par le *Test Pattern Generator* (*TPG*) et ils sont envoyées vers le récepteur R_X. Dans la couche récepteur, les vecteurs de test reçus T_1^*:T_N^* sont comparés avec les valeurs attendues. Les cellules d'analyse des reposes (*RA*) comparent ces valeurs pour générer le vecteur de diagnostique DV_1-DV_N qui indique les TSVs fautives.

Les vecteurs de test qui sensibilisent les fautes de diaphonie peuvent être générés par des TPGs qui implémentent le modèle *Maximal Aggressor Fault* (*MAF*) [CDB99]. Malheureusement, ce modèle est trop conservatif et conduit à de longues séquences de test et à des coûts matériels non négligeables. On propose un nouveau modèle, le K^{th}-*Aggressor Fault* (*KAF*).

Dans les puces 3D les TSVs sont implémentés de façon à ce qu'ils n'affectent pas le fonctionnement des autres composantes. Les interconnections entre couches ont une distribution uniforme ou irrégulière sur des matrices *MxM*. Dans notre modèle, on considère que, pour un pitch *p* et un ordre *K*, les agresseurs d'un TSV sont les TSVs les plus proches. Le première (*K=1*), deuxième (*K=2*), et le troisième (*K=3*) ordre d'agresseurs d'une victime sont représenté dans In Figure III-3, pour distributions régulières et aléatoires.

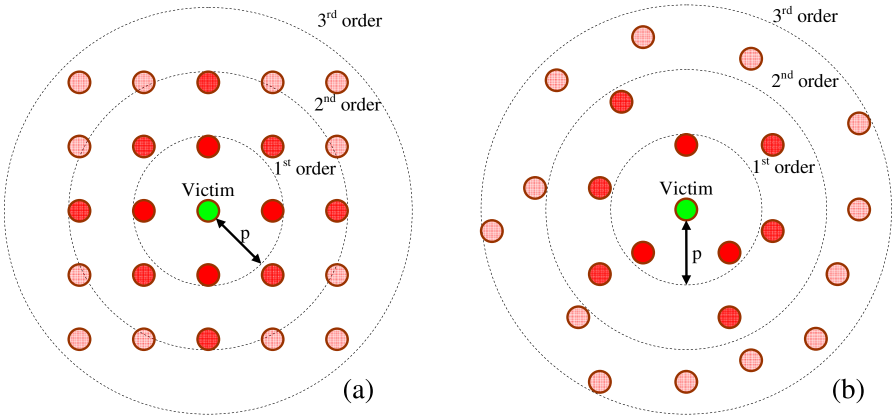


Figure VIII-3 Les 1er, 2ème et 3ème agresseurs d'un TSV dans une distribution reguliére (a) et aléeatoire (b)

L'ordre K pour l'agresseur dépend de la technologie et est déterminé de telle sorte que les temps de test soient minimes et que la couverture de fautes soit maximale. En utilisant le modèle *KAF* avec un ordre K, on suppose que les TSVs sont partagés dans w sets sets $\{V_1, V_2, \ldots, V_w\}$. Pendant le test, les TSV du set V_i sont victimes et les autres TSVs sont les agresseurs. Donc, pour les w sets, les signaux victime P_V sont envoyés sur les TSVs du groupe Vi et les signaux P_A sur les autres TSVs. Pour l'architecture BIST, le circuit qui génère les vecteurs de test est représenté dans Figure III-7.

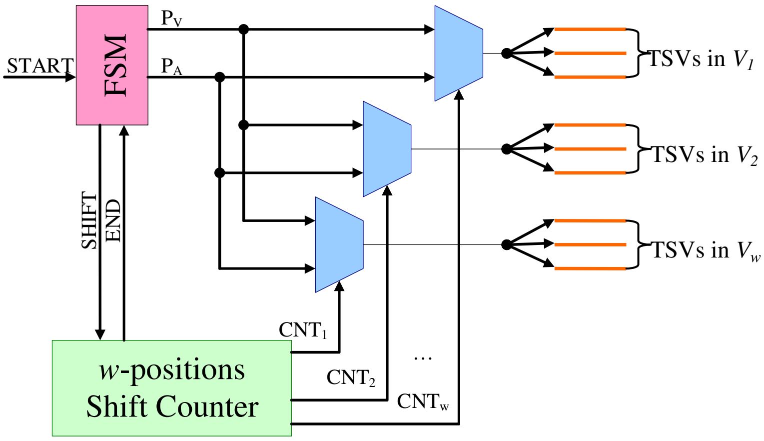


Figure VIII-4 Generateur de Traffic pour IBIST avec le modèle KAF

Dans cette implémentation, le compteur à w positions CNT_1-CNT_w est utilisé pour envoyé les signaux P_V sur les TSVs victimes. Pour chaque groupe V_i, un multiplexeur 2:1 contrôle lesquelles des signaux victimes ou agresseurs sont envoyé sur les TSVs. Une mise en œuvre d'un *IBIST* configurable où les tests peuvent être effectués en utilisant différents ordres d'agresseurs K est également possible.

Les évaluations expérimentales ont montré que la durée du test en utilisant un *BIST* avec des TPGs basés sur le modèle KAF est réduite jusqu'à 16 cycles pour $K=1$. Les évaluations du coût on montré également que la surface du BIST est jusqu'à trois fois plus petite par rapport aux implémentations des TPGs avec des modèles existants comme le *MAF* ou le *Marching*.

Le test des composants intra-die et inter-die des NoCs 3D ne peut pas garantir l'absence de fautes. Toutefois, ce n'est pas suffisant pour assurer une communication fiable. Les fautes transitoires ne peuvent pas être diagnostiquées lors des tests. Les erreurs affectant les messages échangés sur le NoC pourraient avoir des effets dramatiques sur le comportement du système. Dans la section suivante des techniques de tolérance aux fautes pour les NoCs 3D sont présentées.

8.3 *Tolérance aux fautes multi niveaux dans les NoCs 3D*

Dans les NoCs 3D, la fiabilité et les défis du rendement dues à des technologies peu fiables TSV sont adressées à deux niveaux d'abstraction: *data-link* et *network*. Même si des solutions aux niveaux *data-link* et *network* sont capables de corriger les fautes des TSVs, leur efficacité est souvent contrebalancée par les coûts élevés. Donc, une approche multi niveaux, qui s'appuie sur des données des solutions de liaison, au niveau du réseau, est proposée.

8.3.1 ***Tolérance aux fautes au niveau data-link***

Dans cette sous-section des stratégies pour les fautes transitoires et permanentes des TSVs sont présentées. La résistance contre les fautes transitoires et permanentes sur les liens *inter-die* est assurée par l'utilisation de la redondance et des systèmes de codages.

8.3.1.1 Fautes transitoires

La résistance face aux transitoires sur les liens est assurée par l'utilisation de la redondance matérielle et des systèmes de contrôle d'erreur. Du côté émetteur, les données (*flits*) sont codées avant d'être envoyées sur les fils physiques (*PHY*). Du côté récepteur, les *flits* codées reçues sont vérifiées pour les erreurs de transmission. Si des erreurs sont détectées, elles sont traités par le mécanisme de reprise, qui est habituellement mis en œuvre par des moyens de correction d'erreur ou de retransmission. Le contrôle de flux des signaux assure le comportement NoC correcte. Des fautes transitoires sur ces signaux sont plus susceptibles de conduire à une défaillance du système. Par conséquent, les stratégies de protection agressive telles que la redondance modulaire triple (TMR) sont utilisées pour ces signaux.

La technique *Forward Error Correction* (FEC) a la capacité de corriger une ou plusieurs erreurs de transmission. Ces capacités sont données par la puissance de correction du code ou sa distance de Hamming. Pour deux routeurs T_X et R_X, le schéma du lien avec FEC est donné dans Figure IV-1.

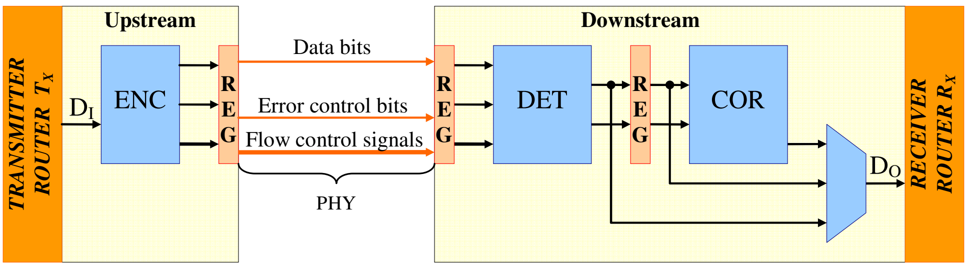


Figure VIII-5 Schéma de la strategie Forward Error Correction (FEC)

Du côté du routeur émetteur T_X, les *flits* sont encodés par le module *ENC*. Après le codage, les bits de données et de parité sont envoyés sur les TSVs. Du côté du routeur récepteur R_X, le module *DET* vérifie s'il y a eu des fautes de transmission. Toutes les erreurs détectées sont corrigées par le module *COR*, avant que les *flits* n'arrive à destination.

La retransmission est utilisée dans la stratégie *Automatic Retransmission Query* (*ARQ*) pour masquer les fautes transitoires. Le schéma de la technique *ARQ* est représenté dans Figure IV-5 et est aussi nommée retransmission *go-back-N*.

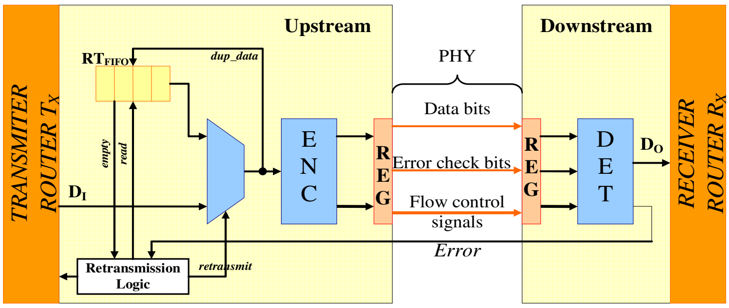


Figure VIII-6 Schéma go-back-N Automatic Retransmission Query (ARQ)

Du côté de l'émetteur T_X, les *flits* sont encodés par le module *ENC* et mémorisés dans des mémoires tampon de retransmission RT_{FIFO}. Les fautes transitoires sont détectées par le module *DET* qui est implémenté du côté du récepteur R_X. Une requête de retransmission est envoyée à l'émetteur, si des erreurs sont détectées, et les *flits* fautés sont renvoyés.

La stratégie hybride de correction et retransmission (*HYB*) utilise les deux techniques présentées avant : *FEC* et *ARQ*. Donc, les *flits* sont codés et mémorisés du côté de l'émetteur T_X et la retransmission est demandée si les erreurs détectées du coté du récepteur ne peuvent pas être corrigées. Le schéma de la stratégie *HYB* est présenté dans Figure IV-7.

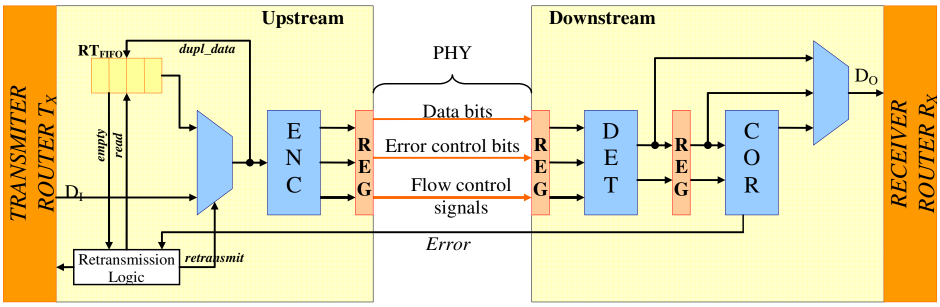


Figure VIII-7 Schéma hybride avec correction et retransmission

Les stratégies de tolérance aux fautes présentées sont implémentées au niveau des liens pour améliorer la fiabilité du réseau. On obtient une fiabilité maximale pour la stratégie où tous les liens sont protégés. Par contre, il n'est pas toujours nécessaire d'implémenter ces stratégies pour tous les liens. Pour les NoCs 3D les coûts de la tolérance aux fautes au niveau *data-link* sont réduits avec l'utilisation d'une stratégie de protection sélective des liens. Donc, seulement les liens verticaux sont protégés.

8.3.1.2 Fautes permanentes

Malgré les capacités de correction des techniques présentées dans la section précédente, des stratégies plus adaptées sont proposées pour les fautes permanentes des TSVs. Après le packaging, des stratégies de test (par exemple *Boudary Scan*) sont utilisées pour identifier les TSVs fautés. En utilisant une structure de réparation sur puce et des TSVs redondants, les TSV fautés sont replacées par des voisins fonctionnels. Donc, la réparation des liens est assurée avec des stratégies de redondance matérielle ou de sérialisation.

Pour la stratégie de réparation des *n* TSVs avec *r* TSV redondants, le schéma du lien inter-die est donné dans Figure IV-11.

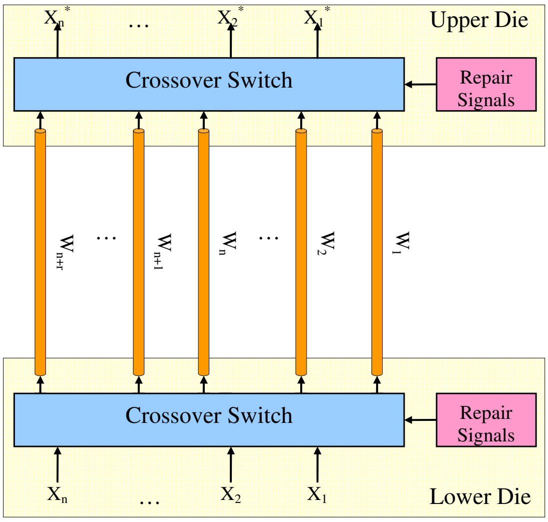


Figure VIII-8 Structure de réparation pour n TSVs avec r fils redondants

Les signaux X_1-X_n se propagent à partir de la couche inférieure vers la couche supérieure, en traversant les structures de réparation (Crossover Switch). Dans la couche inférieure, ces structures envoient les signaux signals X_1-X_n sur les premiers *n* TSVs fonctionnels. Les *switches* sont implémentés comme des matrices de commutation dont les signaux de commande sont les signaux de réparation qui indiquent comment les TSVs fautés sont remplacés. Un possible implémentation d'un switch est donné dans Figure IV-12.

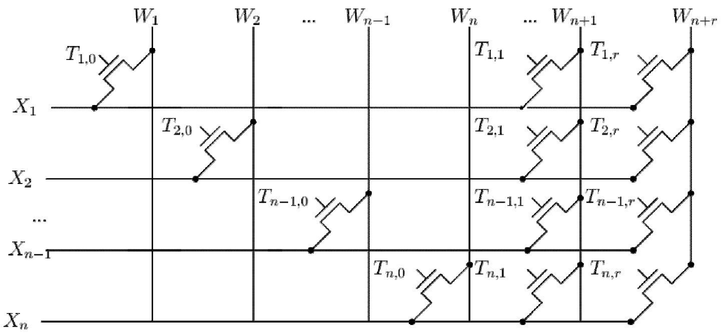


Figure VIII-9 Implémentation du crossover switch avec n entrées et n+r sorties

Les coûts de la stratégie *TSV-SnR* sont donnés par la surface des TSVs redondants et la surface des structures de réparation. Même s'il y a des stratégies pour optimiser le coût, il n'est pas toujours possible d'assurer la réparation des liens. Donc, une stratégie de sérialisation est proposée dans la suite.

Pour un lien, on assume N *TSV* réguliers et R *TSV* redondants. Le lien est fonctionnel si au moins M_{MIN} fils ne sont pas fautés. Donc, on a besoin de jusqu'au $K_{MAX}=\lceil N/M_{MIN} \rceil$cycles pour la transmission sérielle des N bits de données. Le schéma d'un lien *Configurable Serial Link (CSL)* est présenté dans Figure IV-14.

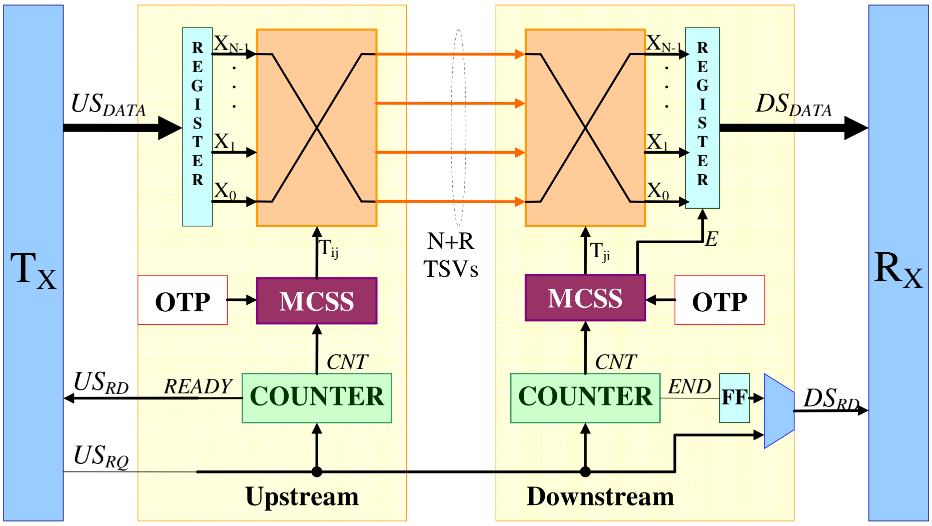


Figure VIII-10 Implémentation du CSL pour N bits

Quand le nombre des TSVs fonctionnels M est inférieur à N, le lien sérialise la transmission dans $K=\lceil N/M \rceil$ cycles. Du côté de l'émetteur, les N bits sont chargés dans un registre et pendant les K cycles de transmission jusqu'à M bits data sont envoyés sur les TSVs fonctionnels. Les signaux reçus du côté récepteur sont mémorisés dans un registre où le message entier est recréé.

Cette technique, aussi que la réparation basée sur la redondance, utilisent une stratégie de test *off-chip* (Boundary Scan) pour identifier les TSVs fautifs. Les signaux de réparation son calculé par le testeur externe est ils sont programmées dans des mémoires fusibles (*one-time-programable*) via la chaîne de scan. Une implémentation *on-chip* est aussi possible. Dans ce cas, la stratégie de test et la partie de calcul des signaux de réparation sont embarquées aux interfaces des liens.

Dans cette section des stratégies *data link* pour les fautes transitoires et permanentes ont été proposées. Dans la suite, plusieurs solutions au niveau *network* son présentées.

*8.3.2 **Tolérance aux fautes au niveau network***

Des techniques de tolérance aux fautes sont implémentées aux niveaux plus haut. Dans cette section on propose des stratégies de tolérance aux fautes transitoires et permanentes au niveau *network*.

8.3.2.1 Fautes transitoires

Dans un NoC, les interfaces construisent des paquets qui sont envoyés sur le réseau *flit par flit*. Donc, on peut modéliser le réseau comme une boîte noire avec des entrées et des sorties connectées aux interfaces réseau. Dans cette section, les défis de fiabilité de la communication sur le NoC 3D sont résolus par la stratégie de correction *network-level Forward Error Correction* (*NL-FEC*), voir Figure IV-24.

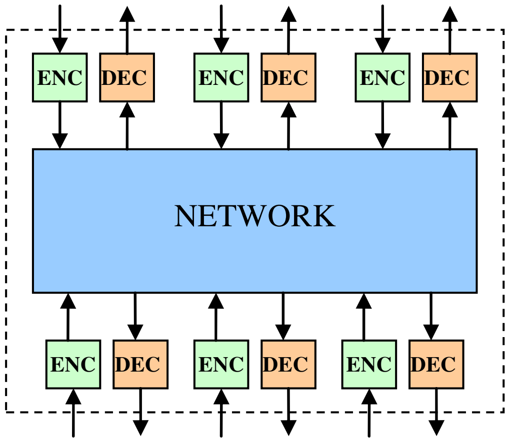


Figure VIII-11 Network-Level Forward Error Correction

Dans *NL-FEC*, les *flits* sont codées aux noeuds source avec un code correcteur d'erreurs comme Hamming ou Hsiao. Du côté des noeuds récepteur, les erreurs de transmission sont détectées et corrigées. Dans les réseaux plus grands, les chemins sont plus longs et la probabilité d'avoir des fautes multiple à destination est non négligeable. Dans ce cas, des codées correcteurs capable de corriger des fautes multiples sont utilisées. Pour *NL-FEC*, on propose d'utiliser plusieurs codes SEC pour chaque partie des *flits*.

8.3.2.2 Fautes permanentes

Dans un réseau, les *paquets* sont dirigés de la source vers la destination sur les chemins indiqués par l'algorithme de routage. Quand il y a des composants fautés dans le réseau, l'algorithme de routage doit trouver des chemins alternatifs, autour de ces composants. Pour les NoCs 3D avec une topologie mesh, on propose un algorithme de routage tolérant aux fautes capable de diriger les données autour des liens 3D fautés.

Les chemins alternatifs qui évitent les liens avec des TSvs fautés ne sont pas minimaux, car ils contiennent des noeuds intermédiaires qui ont des liens verticaux fonctionnels (master nœuds). Pour une topologie mesh 3D, quatre ports du routeur sont pour la communication horizontale dans les directions *NORTH*, *SOUTH*, *EAST* et W*EST*, et deux ports sont pour la communication verticale dans les directions *UP* et *DOWN*. Pour chaque nœud il y a un registre pour les coordonnées des nœuds *master-up* (X_{UP}, Y_{UP}) et *master-down* (X_{DOWN}, Y_{DOWN}). L'algorithme de routage pour le nœud (X_{LOCAL}, Y_{LOCAL}, Z_{LOCAL}) est représenté dans Figure IV-26.

```
if (ZLOCAL = ZDEST) then
    if (YLOCAL = YDEST) then
        if (XLOCAL = XDEST) then OUTPUT(LOCAL);
        elsif (XLOCAL > XDEST) then OUTPUT(NORTH);
        else OUTPUT(SOUTH);
        end if;
    elsif (YLOCAL > YDEST) then OUTPUT(EAST);
    else OUTPUT(WEST);
    end if;
elsif (ZLOCAL > ZDEST) then
   if(YLOCAL = YDOWN) then
      if(XLOCAL = XDOWN) then OUTPUT(DOWN);
      elsif (XLOCAL > XDOWN) then OUTPUT(NORTH);
      else OUTPUT(SOUTH);
      end if;
   elsif (YLOCAL > YDOWN) then OUTPUT(EAST);
   else OUTPUT(WEST);
   end if;
else
  if(YLOCAL = YUP) then
      if(XLOCAL = XUP) then OUTPUT(UP);
      elsif (XLOCAL > XUP) then OUTPUT(NORTH);
      else OUTPUT(SOUTH);
      end if;
   elsif (YLOCAL > YUP) then OUTPUT(EAST);
   else OUTPUT(WEST);
   end if;
end if;
```

Figure VIII-12 Algorime de routage tolerant aux faultes pour les topologies mesh 3D

Dans cet algorithme on utilise le routage horizontal YX pour arriver aux nœuds *master-up* si $Z_{LOCAL}<Z_{DEST}$, *master-down* si $Z_{LOCAL}<Z_{DEST}$, *ou destination* si $Z_{LOCAL} = Z_{DEST}$. L'algorithme est réutilisé par chaque nœud dans le chemin source-destination du paquet. Pour illustrer le fonctionnement de l'algorithme, on prend dans Figure IV-27 plusieurs cas avec et sans des liens fautés.

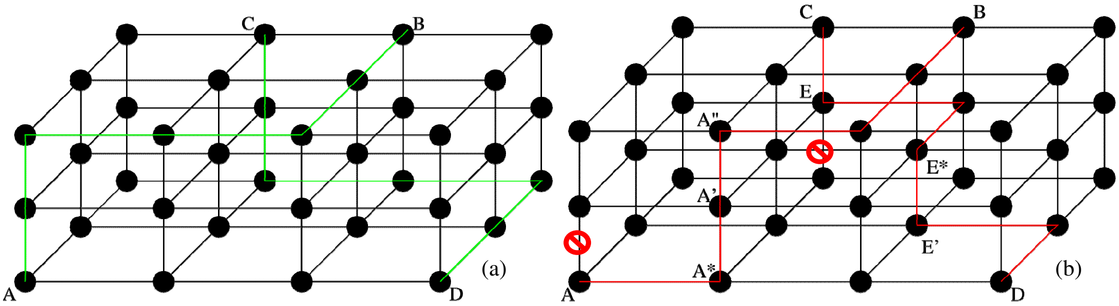


Figure VIII-13 3x4x3 3D mesh topology

Dans le chemin *A-to-B*, le lien UP du routeur A est fauté. Les paquets de A à B sont donc envoyé d'abord à A*, le master-up du routeur A. Après ils continuent leur chemin vers le nœud B via *A'* et *A''*. Pour le chemin *C-to-D* on a des soucis car le lien *DOWN* du routeur *E* est fauté. Les paquets arrivant a E sont alors renvoyé au nœud E*, le *master-down* du routeur *E*.

8.3.3 Stratégies de tolérance aux fautes sur multiple niveaux

Les stratégies proposées dans la section précédente essaye de masquer les fautes des TSVs a un seul niveau d'abstraction. Il est aussi possible de partager cette tâche sur plusieurs niveaux. Dans cette section on propose deux stratégies de tolérance aux fautes pour le NoCs 3D qui utilisent les niveaux data link et network.

8.3.3.1 Fautes transitoires

Dans la stratégie de protection multicouche, des codes de corrections d'erreurs sont utilisées aux niveaux *data link* et *network*. Au niveau du réseau, les flits sont codés avec des ECC. La détection et la correction des erreurs sont effectuées au niveau network mais aussi au niveau des liens intermédiaires. Donc, la fiabilité est augmentée, car les fautes se cumulent entre deux étages de correction et non sur tout le chemin. Pour un NoC 3D, ces étages de correction sont implémentés sur les liens verticaux, voir Figure IV-32.

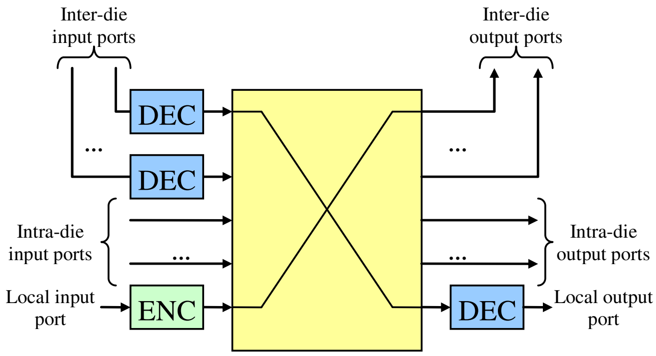


Figure VIII-14 Implementation du NL-FEC avec correction au niveau des liens verticaux

Si la fiabilité du réseau n'est pas suffisante, des étages de correction sont implémentés au niveau des liens *intra-die*. Pour un *mesh 2D* avec un nombre maximal de sauts fiables p_{MAX}, on peut déterminer quelles sont les liens qu'il faut protéger. Dans Figure IV-33, six étages de correction sont rajoutés pour un mesh 4×4 avec $p_{MAX}=4$.

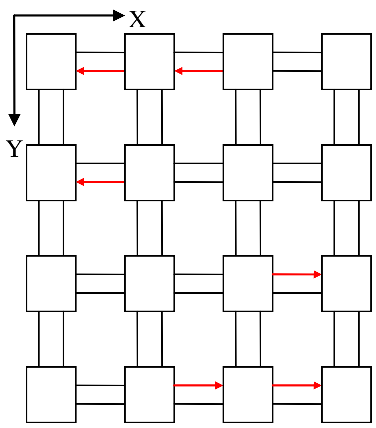


Figure VIII-15 Stages de correction pour un mesh 4×4 avec $p_{MAX}=4$

Même si une telle stratégie est suffisante pour les fautes permanentes des TSVs, il y a des cas où les capacités de correction des fautes sont très limitées. Donc, une stratégie multi niveau pour les fautes permanentes est présentée dans la suite.

8.3.3.2 *Fautes permanentes*

La stratégie multicouche est utile pour les solutions de réparation des TSVs avec des fils redondants ou la sérialisation, mais aussi l'algorithme de routage tolérant aux fautes. Après le final, les puces 3D avec des composantes *intra-die* fautés ou irréparable sont jetées (*Known-Good-Die* testing). Donc, on a des systèmes qui contiennent seulement des TSVs fautés. Le diagramme de cette stratégie multi niveau est présentée dans Figure IV-31.

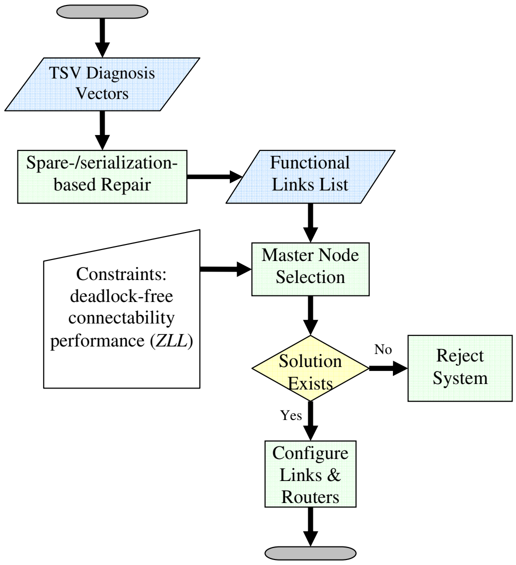


Figure VIII-16 Strategie de réparation multicouche pour les NoCs 3D

On commence d'abord avec les réparations au niveau du lien. Après cette phase, on a la liste des liens verticaux fonctionnels et fautés. Pour chaque routeur avec un lien *inter-die* fauté, il faut lui trouver un *master* dans la direction du lien. Seulement si on trouve une solution pour tous les nœuds on peut configurer le NoC en programmant les mémoires *one-time-programmable* (*OTP*) des routeurs et des liens.

Dans cette section, on a présenté plusieurs stratégies pour les fautes transitoires et permanentes des TSVs. Pour chaque stratégie, on peut trouver une configuration qui permet de tolérer un grand nombre de fautes. Par contre, c'est très difficile d'évaluer les coûts et l'impact sur les performances. Dans la suite, on présentera le sommaire des résultats expérimentaux pour le NoC 3D avec une topologie mesh régulière.

8.4 *Résultats expérimentaux*

Les techniques de tolérance aux fautes sont implémentées pour le routeur sur les sept ports des NoCs 3D avec une topologie mesh. Dans cette section, les coûts et l'impact sur les performances et fiabilité du réseau sont déterminés pour les stratégies *data link* et *network*.

8.4.1 ***Fautes transitoires***

Au niveau *data link*, les techniques *FEC*, *ARQ* et *HYB* sont utilisées pour améliorer la fiabilité des liens. Pour chaque méthode on peut trouver la configuration optimale pour garantir une fiabilité. Dans la section précédente, on a montré que il est possible de réduire les coûts des stratégies *data link* en utilisant la méthodologie de protection sélective des liens verticaux. Pour la technologie de 65nm, les stratégies sont implémentées pour une fréquence de *1GHz*. Les coûts relatifs en surface et puissance sont donnés dans TABLE V pour des routeurs avec *32* et *64* bits.

TABLE V COÛTS RELATIFS EN SURFACE ET PUISSANCE POUR LE ROUTEUR AUX SEPT PORTS AVEC LES DEUX STRATÉGIES DE PROTECTION

Strategie	32 Bits				64 Bits			
	Tous liens		Liens Inter-die		Tous liens		Liens Inter-die	
	Surf. (%)	*Puiss. (%)*	*Surf. (%)*	*Puiss. (%)*	*Surf. (%)*	*Puiss. (%)*	*Surf. (%)*	*Puiss. (%)*
FEC SEC	25.90	35.78	8.46	11.22	26.7	35.32	8.73	11.51

FEC SEC×2	26.98	38.36	9.09	12.12	27.39	37.43	8.9	12.02
FEC SEC×4	28.11	43.22	9.41	13.23	27.95	40.34	9.16	13.31
ARQ CRC-5	36.65	46.96	10.2	13.45	39.9	48.6	11.25	14.17
ARQ CRC-8	38.17	48.44	10.71	13.50	44.52	48.78	11.13	14.26
ARQ PARITY×4	42.85	57.21	12.16	16.44	45.64	43.07	16.35	10.63
HYB SECDED	77.15	95.49	21.34	27.58	83.56	95.35	22.98	27.97
HYB SECDED×2	77.16	103.11	21.27	29.32	81.57	99.03	22.62	29.31

Ces résultants montre que les stratégies de protection sélective sont jusqu'à trois fois moins chère que la stratégie de protection pour tous les liens. Parmi les stratégies *data link*, on trouve que les coûts en surface et puissance de la stratégie *FEC* sont les plus basses.

Le fait de ne protéger que les liens verticaux a un impact sur la fiabilité des chemins source-destination dans le réseau. Pour un chemin comprenant 12 nœuds avec $n_h=9$ liens *intra-die* et $n_v=2$ liens *inter-die,* la fiabilité est représentée dans Figure V-11 pour des liens avec *32* et *64* bits.

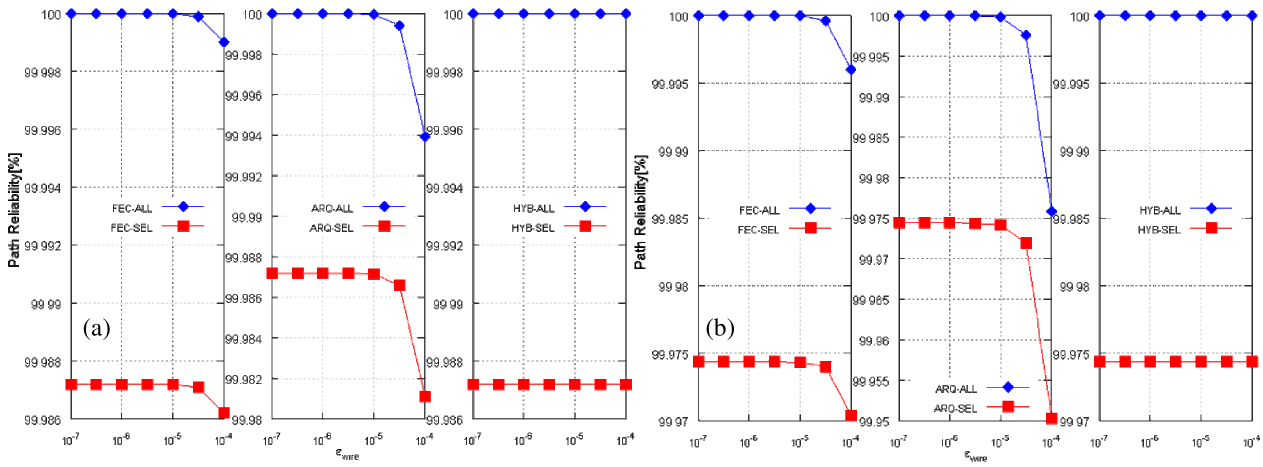


Figure VIII-17 Fiabilité du réseau avec tolerace aux fautes sélective (SEL) et compléte (ALL) pour un taux de defaillance de $\varepsilon_{wire}=10^{-6}$ pour les liens intra-die

Les évaluations ci-dessous montrent qu'une baisse de la fiabilité de *0.012%* pour les liens de 32 bits et de *0.025%* pour les liens de 64. Il faut noter que la variation de la fiabilité avec le taux de défaillance est négligeable pour la stratégie *HYB*. Donc, la stratégie de protection sélective est efficace pour les applications ayant des objectifs de fiabilité plus faible. Cette stratégie est aussi utile dans les réseaux avec des liens verticaux plus susceptibles aux fautes.

Au niveau *network*, les modules de codage et de décodage sont implémentés au niveau du port *LOCAL* des routeurs. Les coûts relatifs de la stratégie *NL-FEC* avec les différents schémas de codage correcteur d'erreurs sont présentés dans TABLE VI pour les routeurs aux *32* et *64* bits dans la technologie de *65 nm*.

TABLE VI COÛTS RELATIFS POUR LE ROUTEUR AVEC NL-FEC

Error correction scheme	32-bits		64-bits	
	Surf (%)	Puiss (%)	Surf (%)	Puiss (%)
Hamming SEC × 1	19.81	15.55	18.84	9.75
Hamming SEC × 2	33.61	30.61	24.93	16.62
Hamming SEC × 4	49.88	48.78	37.72	28.98

Les résultats montrent que les coûts augmentent avec la complexité du code correcteur. Ces coûts sont dus aux mémoires tampon d'entrée et de sortie des routeurs qui représentent environ 90% de leur surface. Pour une stratégie de codage avec plusieurs groupes, la taille des *flits* augmente et donc la taille des routeurs augment aussi. Une stratégie de codage plus complexe est nécessaire dans les réseaux où la probabilité d'avoir des fautes multiples est non négligeable. Pour les routeurs à 64 bits, les coûts relatifs sont moins importants car le nombre relatif de bits de parité est inférieur.

Dans la stratégie multi niveau pour les fautes transitoires, des étages de corrections sont rajoutés au niveau des interfaces routeurs-liens. Pour le routeur 3D mesh, ces étages de correction sont implémentés pour jusqu'à six ports. Pour les routeurs avec 32 et 64 bits, les coûts relatifs sont présentés dans TABLE VII pour la technologie de 65 nm.

TABLE VII COÛTS RELATIFS EN SURFACE ET PUISSANCE POUR LE ROUTEUR AVEC LA STRATEGIE MULTICOUCHE

Error correction scheme	32-bits		64-bits	
	Surf (%)	Power (%)	Area (%)	Power (%)
SEC avec 1 port	27.64	23.19	26.75	17.83
SEC avec 2 ports	32.43	29.01	31.83	23.81
SEC avec 3 ports	34.54	30.25	37.77	29.57
SEC avec 4 ports	40.15	39.79	43.02	35.28
SEC avec 5 ports	45.65	46.3	50.86	42.12
SEC avec 6 ports	48.02	48.84	53.97	45.55

Les résultats montrent que les coûts augmentent avec le nombre d'étages de correction implémentée. Ces coûts sont dus à la taille des *flits* et aussi aux modules de correction d'erreurs qui augmentent aussi le nombre de *flits* contenus dans les buffers d'entrée. Pour les NoCs 3D mesh, les étages de correction sont rajoutés pour les deux liens verticaux. Donc, par rapport aux stratégies NL-FEC avec les codes de correction multiples, on a un petit avantage au niveau de la surface et la puissance des routeurs. Un autre avantage est le nombre réduit des TSVs nécessaires pour envoyer les *flits* codés.

8.4.2 ***Fautes permanentes***

Au niveau *data link* les stratégies de réparation avec les TSV redondants et la sérialisation sont implémentées pour les deux liens verticaux du routuer à sept ports. Cette stratégie à des avantages ainsi que des limitations, et dans la suite une comparaison de ces stratégies est présentée pour les différentes technologies.

Pour le routeur avec 32 data bits, les stratégies *SnR* and *CSL* sont configurées pour un rendement objectif Y_T*=99.95%*. Pour analyser l'impact des technologies TSV, on diminue la distance minimale des TSVs de *15µm* à *10 µm*. Donc, le nombre des TSVs est doublé et le nombre ainsi que la taille des liens augmentent.

Pour ces configurations, la surface relative est représentée dans Figure V-5 pour les différents taux de défaillance.

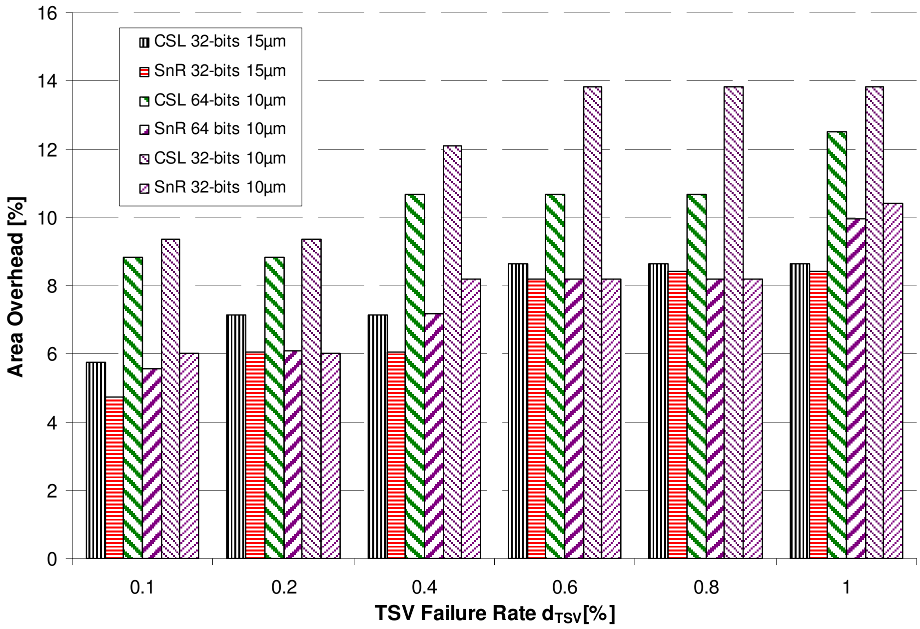


Figure VIII-18 Coûts rélatifs en sourface pour le routeur avec TSV-SnR et CSL

Les résultats montrent que la solution *TSV-SnR* est plus efficace que la sérialisation en termes de surface. Les coûts sont donnes par la surface des circuits de réparation et TSV. Donc, pour la technologie de *65nm,* les différences entre les deux solutions ne sont pas constantes. *P*our les TSVs avec un pitch de *15µm*, les différences diminuent avec le taux de défaillance. Donc, on peut extrapoler ces résultats et dire que dans les technologies avec des grands taux de défaillance, la sérialisation est plus efficace, car le nombre des ressources redondantes est trop élevé.

Au niveau réseau, la réparation des TSV fautés est faite par l'algorithme de routage tolérant aux fautes. Les fautes sont réparées si on trouve les noeuds master pour chaque noeud avec des liens inter-die fautés. Pour le routeur 3D mesh qui implémente le routage ZYX, deux registres sont rajoutés pour mémoriser les coordonnées des nœuds master et deux possibles implémentations de l'algorithme de routage sont proposées dans Figure V-6 (a) et (b).

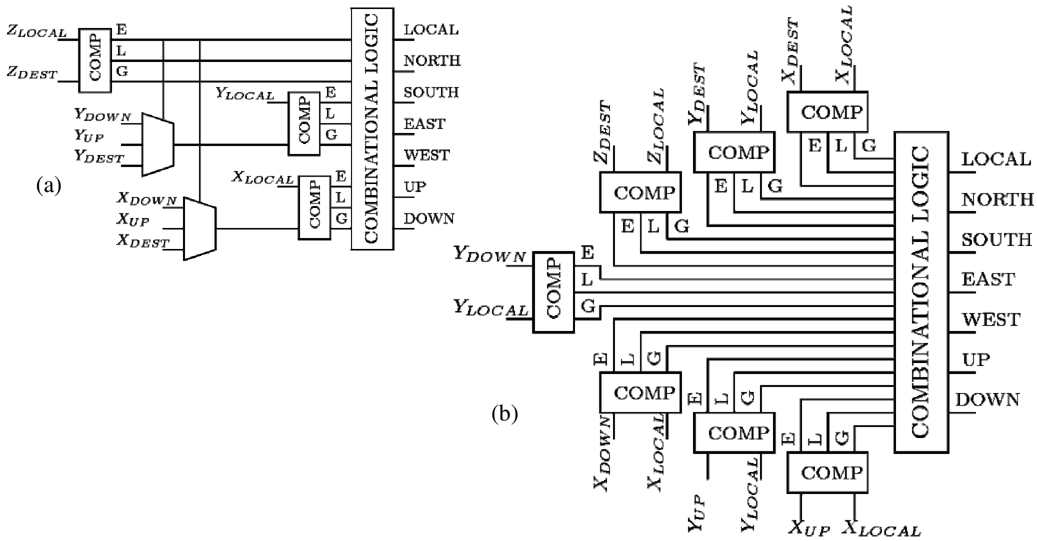


Figure VIII-19 Impémentations du module de routage ZYX pour TSV-FTR

Dans l'implémentation *low-area*, le coût est inférieur à *0.5%* dans la technologies de *65 nm*. Par contre, il y a un impact sur le timing du routeur et la fréquence maximale est *~900MHz*. Dans l'implémentation haute performance l'impact sur le timing est négligeable, mais le coût relatif en surface et en puissance est *~7.7%/5.4%* pour 32 bits et *6.18%/4.9%* pour 64 bits.

Dans la stratégie multi niveau toutes les stratégies *data link* et *network* sont utilisées ensemble. Pour le routeur 3D mesh plusieurs stratégies multi niveau sont implémentées : FRT avec un ou deux TSVs redondants, FTR avec sérialisation et FTR avec sérialisation et un TSV redondant. Les coûts relatifs en surface sont donnés dans la TABLE VIII pour les 32 et 64 bits data et un TSV pitch à *15µm* et *10µm*.

TABLE VIII COÛTS RELATIFS EN SOURFACE POUR LE ROUTEUR AVEC DIFFERENTS STRATÉGIES

Data Size (bits)	p_{TSV} (µm)	FTR+ 1 Spare	FTR+ 2 Spares	FTR+CSL	FTR+ CSL+ 1 Spare
32	10µm	11.89%	13.82%	16.53%	18.65%
	15µm	12.01%	14.95%	16.53%	19.25%
64	10µm	9.62%	11.76%	16.08%	19.95%
	15µm	9.98%	12.33%	16.08%	20.1%

Les stratégies multi niveau sont plus chères que les solutions *data link* ou *network* au taux de défaillance très petits. Par contre, la complexité de *TSV-SnR* en *CSL* augmente avec d_{TSV} et on trouve des configurations où ces solutions sont plus chères. Même si la solution au niveau réseau est la moins chère, ces résultats ne montrent pas l'impact sur la performance du réseau. L'avantage des solutions multi niveau est une possible amélioration de ces performances avec un petit coût matériel.

Pour le NoC 5×5×4, on analyse la latence pour plusieurs configurations résistantes aux fautes permanentes: routage tolérant aux fautes, sérialisation, sérialisation avec *TSV-SnR*, routage tolérant aux fautes avec TSVs redondants et sérialisation verticale. Les résultats sont représentés dans Figure V-7 pour plusieurs taux de défaillance.

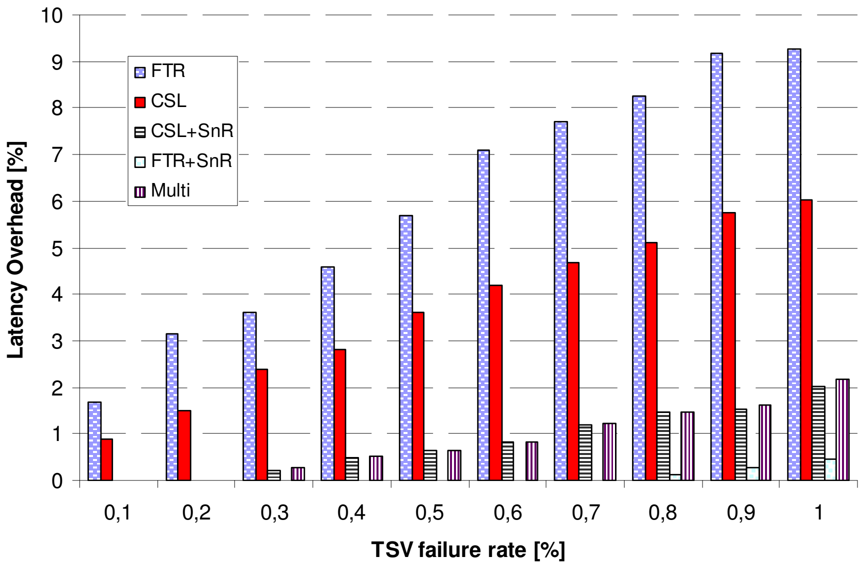


Figure VIII-20 Latence rélative pour un mesh 5x5x4 protégée avec differentes solutions

Pour un taux de défaillance bas, l'impact sur la performance des stratégies utilisant les TSVs redondants est négligeable. Dans ces configurations, la plupart des fautes sont réparées par *TSV-SnR*. Parmi ces configurations, on trouve que le routage tolérant aux fautes avec un TSV redondant est le meilleur. Bien sûr, si on ne peut pas inclure les TSVs redondants pour chaque lien, nos résultats montrent que l'impact de la sérialisation est moins important que celui du routage tolérant aux fautes.

Dans cette section, on a évalué différentes stratégies de tolérance aux fautes transitoires et permanentes des TSVs. On a montré que, dans certains cas, une solution à plusieurs niveaux est plus efficace. Par contre, c'est très difficile d'indiquer la meilleure solution pour un environnent donnée. Donc, une stratégie d'exploration des solutions est présentée dans la section suivante.

8.5 *Flot d'injection de tolérance aux fautes*

Dans cette section on présente une stratégie qui permet d'identifier les configurations tolérantes aux fautes pour les NoCs 3D qui atteint les objectifs de fiabilité et de rendement imposés par le designer. Pour une topologie 3D mesh donnée, les informations sur la fiabilité des composants (défaillance des TSVs, bit-error-rate) sont à utiliser pour trouver la configuration avec un coût et un impact sur la performance inférieur à une limite imposée. Le schéma du processus d'exploration des solutions tolérantes aux fautes pour le NoC 3D mesh est présenté dans Figure VI-1.

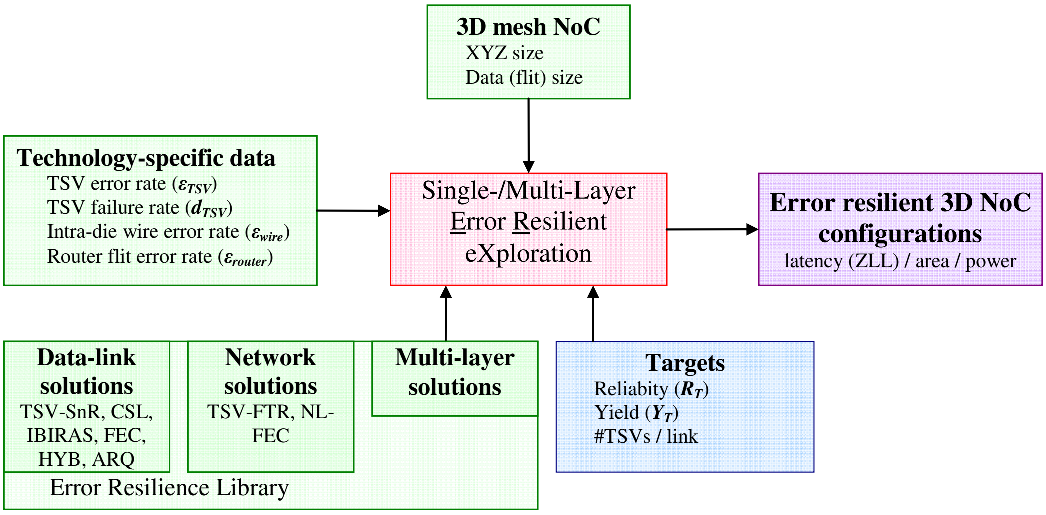


Figure VIII-21 Le processus d'éxploration pous les 3D NoCs

Le processus d'exploration est automatisé et intégré dans un flot de conception des NoCs 3D mesh. Les composantes tolérantes aux fautes sont implémentées dans une bibliothèque générique et configurable: la *Error-Resilience Library*. Cette bibliothèque est compatible avec le flot de conception des NoCs, c'est à dire qu'elle implémente les mêmes protocoles de communication au niveau lien et réseau.

Le processus d'exploration commence par les solutions pour les fautes transitoires. Donc, les configurations pour les solutions *data link*, *network* et *multi niveau* sont élaborées et seulement celles qui tiennent les objectifs de fiabilité avec un nombre de TSVs par lien inférieur à la limite imposée sont gardées. Pour chaque configuration, on essaye de trouver une ou plusieurs solutions pour les fautes permanentes en utilisant les mêmes contraintes au niveau fiabilité, rendement et nombre des TSVs par lien. Le résultat de ce processus est une liste des solutions possibles. En utilisant un flot ASIC, on peut déterminer les coûts pour chaque configuration possible. L'impact sur la performance du NoC est évalué par des méthodes analytiques ou par des simulations, si le flot de conception est prévu à cet effet.

Une fois que toutes ces solutions sont évaluées, c'est le concepteur du système qui va choisir la solution qui lui convient le plus. Pour montrer le fonctionnement du *ERX*, on utilise un système 3D modélisé en SystemC avec la bibliothèque Soclib. Le system contient 64 clusters et dans le cache cluster il y a jusqu'au quatre microprocesseurs. Les clusters sont connectés par un 3D mesh 4×4×4 avec deux réseaux : une pour les requêtes avec un flit de *37* bits et une pour les réponses avec un flit de *33* bits. Pour ce système, on utilise *ERX* sur les deux réseaux pour identifier les configurations qui peuvent atteindre les objectifs de rendement $Y_T=99.99\%$ et fiabilité $R_T=99.9999\%$. Pour les évaluations des coûts, on utilise Synopsys Design Compiler® avec la technologie *65 nm* pour un horloge à *1GHz*.

Dans la configuration du système, les fautes transitoires intra-die ne sont pas prise en compte ($\varepsilon_{router}=0$, $\varepsilon_{wire}=10^{-9}$ et $\varepsilon_{TSV}=10^{-6}$), et le taux de défaillance des TSV est de $d_{TSV}=0.01\%$. Dans ce cas, les flits arrivent correctement à destination avec une probabilité de 99.988% pour le réseau des requêtes et de 99.989% pour le réseau des réponses. Le rendement des TSVs pour ce système est de seulement 49%. Pour chaque lien vertical on peut rajouter jusqu'au 10 TSVs. Par contre, *4* TSVs sont utilisés pour la protection *TMR* des lignes de contrôle *write* et *write_ok* du lien. Les configurations proposées par ERX sont les suivantes.

Dans le premier cas, la protection contre les fautes transitoires sur les liens verticaux est assurée par ARQ avec deux bits de parité. Pour les fautes permanentes, deux TSVs redondants sont suffisants. Les coûts relatifs en surface et en puissance de cette solution sont *14%* et *10%* respectivement pour le réseau de requêtes et *12%* et *9.5%* respectivement pour le réseau de réponses. En utilisant la méthode analytique, la latence des réseaux augmente par *2.5%* pour le réseau des requêtes et par *2.49%* pour le réseau des réponses.

Dans la deuxième configuration, les liens verticaux sont protégés par une stratégie FEC avec un code de Hamming. Pour les défauts de fabrication des TSVs, l'algorithme de routage tolérant aux fautes est utilisé. Donc, les coûts relatifs sont estimés à *19%* et à *12%* pour les routeurs dans le réseau des requêtes et à *18.85%* et à *11.55%* pour les routeurs dans le réseau des réponses. La latence augmente par *2.2%* et *2.17%* pour les deux réseaux.

Dans la dernière configuration, un FEC avec un codage de Hamming est utilisé pour les fautes transitoires sur les liens verticaux et la sérialisation pour les fautes permanentes. Les coûts relatifs en surface et en puissance de cette configuration sont de *19.5%* et *12.3%* pour les routeurs 37 bits et *18.985%* et *11.65%* pour les routeurs à 33 bits. La latence moyenne des réseaux des requêtes et des réponses augmente par *~2%* et *2.1%*.

Toutes les configurations ont un impact négligeable sur la latence du NoC. Donc, la décision peut être prise par rapport aux coûts en surface et en puissance. Parmi les configurations proposées, c'est la première qui est la meilleure. Dans une implémentation au niveau système, les simulations ont montré que l'impact du cette solution sur les performances du système sont négligeables.

8.6*Conclusions*

Dans les technologies CMOS avancées, les problèmes de puissance et de délais des interconnections peuvent être résolus par l'intégration 3D. Au niveau système, les puces 3D délivrent plus de puissance de calcul et sont moins gourmands en énergie. Par contre, connecter des centaines de cours d'une façon efficace nécessite une nouvelle architecture de communication évolutive: le Réseau-sur-Puce 3D (*Network-on-Chip* NoC).

Malgré les avantages des technologies 3D à base de TSVs, ces technologies ne sont pas assez fiables pour une production industrielle. Le test, le coût de fabrication, la fiabilité, le rendement, le management thermique et la dissipation de la chaleur restent les défis majeurs. Dans les 3D MPSoCs avec des connections NoC, l'intégrité de la communication inter-die est vitale. Une seule faute peut causer la panne du système. Donc, il faut impérativement implémenter des stratégies efficaces de test de la tolérance aux fautes. Dans cette thèse des solutions pour ces défis ont été proposées.

Le test des TSVs est très difficile surtout quand la puce contient un nombre élevé (centaines ou milliers) des connections verticales. La stratégie d'autotest *Interconnect Built-In Self-Test* (*IBIST*) a été propose dans cette thèse pour les liens verticaux des NoCs 3D. Pour cette architecture, une nouvelle stratégie de test est également présentée. Parmi les avantages de cet algorithme, la capacité de détecter des fautes structurales, mais aussi les fautes de délai et de diaphonie. En utilisant le modèle agresseur – victime, la stratégie *K^{th} Aggressor Fault (KAF)* est proposée. Les avantages de cette méthodologie sont des tests plus courts et des coûts matériels qui sont jusqu'à trois fois moins importants par rapport aux stratégies existantes. Une implémentation configurable permet d'effectuer les tests avec différents ordres agresseurs *K* est aussi présentée. La configurabilité du IBIST permet une réduction des temps de test *in-field* et permet une calibration du modèle KAF pour une technologie TSV.

Des stratégies, uni et multi niveaux, sont utilisées pour résoudre les problèmes de fiabilité et de rendement des TSVs. Dans les NoC 3D, ces solutions sont implémentées au niveau *data-link* et *network*. Au niveau *data-link*, les fautes permanentes dues aux défauts de fabrication sont réparées par TSVs redondants (*TSV Spare-and-Replace*) et via une stratégie de sérialisation configurable (*CSL*). Ces stratégies sont aussi adaptées pour les fautes de vieillissement des TSVs dans une méthodologie complexe de réparation et de sérialisation : *Interconnect Built-In Self-Repair and Adaptive Serialization* (*IBIRAS*). Au niveau réseau, les liens qui contiennent des TSVs fautés ne sont pas utilisés et un algorithme de routage tolérant aux fautes qui évite ces composantes a été proposé. Dans la stratégie multi niveau, les solutions *data-link* sont utilisées pour réduire l'impact du routage tolérant aux fautes sur les performances du réseau. Des méthodes de contrôle d'erreurs au niveau *data link* sont proposées pour les fautes transitoires sur les TSVs. Ces solutions utilisent des codes pour la détection et la correction d'erreurs et des mécanismes de retransmission. Au niveau réseau, une solution pour corriger les fautes *end-to-end* est proposée. Pour améliorer les capacités de correction de cette stratégie, des étages de correction au niveau *data-link* sont aussi utilisés.

Pour évaluer les coûts de ces stratégies, un flot de conception pour des NoCs 3D synchrones avec une topologie régulière mesh a été développée. Les évaluations montrent que les stratégies traditionnelles, dont notamment la retransmission ou les TSV redondants, sont toujours les meilleurs, surtout pour un taux de défaillance élevé. Donc, des stratégies qui utilisent des codes correcteurs sont plus efficaces pour les fautes transitoires. En ce qui concerne les fautes permanentes dues aux défauts de fabrication, on a montré que, dans le cas où il n'y a pas de TSVs redondants, la sérialisation est plus efficace que le routage tolérant aux fautes. Par contre, une stratégie multi niveau avec routage tolérant aux fautes et TSVs redondants est plus efficace en terme de coûts et d'impact sur la performance.

Malgré le nombre de solutions proposées, il est difficile de trouver la solution optimale pour une configuration donnée. A cet effet, un outil d'exploration des solutions de tolérance aux fautes (*ERX*) est intégré dans le flot de conception des NoCs 3D. L'outil prend en entrée, la configuration du NoC 3D (topologie et taille du flit), les objectifs de fiabilité et rendement, et les taux de défaillance des composants. Cet outil donne en sortie une liste des solutions possibles pouvant être utilisée pour instancier les NoC 3D tolérants aux fautes et les analyser avec des flots de conception *ASIC* existantes. Après les évaluations, c'est au concepteur du système de choisir et d'implémenter la meilleure solution.

Dans une étude pour un système massivement parallèle, *ERX* a été utilisé pour identifier la solution optimale. Le système est modélisé par des composants de la bibliothèque SocLib qui est implémenté en *cycle-*

accurate SystemC. Pour un système à 64 clusters interconnectés par un mesh 4 × 4 × 4, les solutions optimales ont été trouvées pour deux configurations. Au niveau système, les simulations ont montré que l'impact des ces solutions sur les performances sont négligeables.

Les travaux présentés dans cette thèse ne sont pas complets et plusieurs directions de recherche sont envisageables. Il est toujours possible d'implémenter d'autres solutions de tolérance aux fautes au niveau data et network, mais aussi aux autres niveaux d'abstraction. Un autre possible direction de développement concerne le flot de conception du NoC 3D. Dans cette thèse, les évaluations des coûts sont faites avec un flot ASIC pour les technologies 2D. Il est possible d'inclure dans ce flot les technologies 3D et les TSVs. Les mesures de performances des NoCs ont été faites avec des formules analytiques ou par simulations en utilisant le modèle de trafic uniforme aléatoire. Pour des évaluations plus précises, il est possible d'introduire des modules d'injection du trafic plus complexes ou même du trafic réel.

References

[AN00] L. Anghel, M. Nicolaidis. *Cost reduction and evaluation of temporary faults detecting technique*. In Proceedings of DATE, March, 2000.

[AN05] D. Avresky, N. Natchev. Dynamic reconfiguration in computer clusters with irregular topologies in the presence of multiple node and link failures. IEEE Transactions on Computers, no. 54, vol. 54, pp. 603–615, 2005.

[AT03] A. Jantsch , H. Tenhunen. Networks on chip. Kluwer Academic Publishers, Hingham, MA, 2003.

[BAC 07] K. Bernstein, P. Andry, J. Cann, et al. *Interconnects in the third dimension: design challenges for 3D ICs*. Proceedings of the 44th Design Automation Conference (DAC), pp. 562-567, 2007.

[BBM05] D. Bertozzi, L. Benini, G. De Micheli. *Error control schemes for on-chip communication links: the energy-reliability tradeoff*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems , vol.24, no.6, pp. 818-831, June 2005.

[BdM02] L. Benini, G. de Micheli. *Networks on Chips: A New SoC Paradigm. Computer*, vol. 35, no. 1, p.70-78, January 2002.

[BMR08] T. Brunschwiler, B. Michel, H. Rothuizen, U. Kloter, B. Wunderle, H. Oppermann, and H. Reichl. *Interlayer cooling potential in vertically integrated packages*. Microsystem Technology vol. 15, no. 1, pp. 57-74, 2008.

[BSS08] A. Bartzas, K. Siozios, D. Soudris. *Three Dimensional Network-on-Chip Architectures*. Chapter 2 in Networks-on-Chips: Theory and practice. CRC Press, 2008.

[CA95] C. M. Cunningham, D. Avresky. *Fault-tolerant adaptive routing for two-dimensional meshes*. In Proceedings of the IEEE Symposium on High-Performance Computer Architecture, page 122, 1995.

[CAA09] A. K. Coskun, J. L. Ayala, D. Atienza, T. Simunic Rosing, and Y. Leblebici. *Dynamic thermal management in 3D multicore architectures*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 1410-1415, 2009.

[CAR10] A. K. Coskun, D. Atienza, T. S. Rosing, T. Brunschwiler, and B. Michel. *Energy-efficient variable-flow liquid cooling in 3D stacked architectures*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 111-116, 2010.

[CC01] Chun-Lung Chen, Ge-Ming Chiu. *A fault-tolerant routing scheme for meshes with nonconvex faults*. IEEE Transactions on Parallel and Distributed Systems, vol. 12, no. 5, pp. 467–475, 2001.

[CDB99] M. Cuviello, S. Dey, X. Bai, and Y. Zhao. *Fault Modeling and Simulation for Crosstalk in System-on-Chip Interconnects*. Proceedings of International Conference on Computer-Aided Design (ICCAD), pp. 297 – 303, 1999.

[CGL08] M. Coppola, M.D. Grammatikakis, R. Locatelli, G. Maruccia, L. Pieralisi. *Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC*. CRC Press, Inc. 2008.

[CNP07] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, V. Narayanan, M. S. Yousif, C. R. Das. *A novel dimensionally-decomposed router for on-chip communication in 3D architectures*. Proceedings of the 34th annual international symposium on Computer architecture (ISCA). pp. 138-149, 2007.

[Cos11] A. Coskun, et al. *Thermal Modeling and Management of Liquid-Cooled 3D Stacked Architectures*. VLSI-SoC: Technologies for Systems Integration, p. 34-55, 2011

[CPB06] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, M. Orshansky. *BulletProof: a defect-tolerant CMP switch architecture*. Proceedings of International Symposium on High-Perfomance Computer Archiecture (HPCA), pp. 5-16, 2006.

[DFB12] A. DeOrio, D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, J. Hu, and G. Chen. *A Reliable Routing Architecture and Algorithm for NoCs.* Transactions on CAD, vol. 31, no. 5, pp. 726-739 2012.

[DM03] T. Dumitras and R. Marculescu. On-chip stochastic communication. In Proceedings of the conference on Design, Automation and Test in Europe, DATE, page 10790-10795, 2003.

[DSV11] F. Darve, A. Sheibanyrad, P. Vivet, F. Petrot. *Physical Implementation of an Asynchronous 3D-NoC Router Using Serial Vertical Links*. Proceedings of Symposium on VLSI (ISVLSI), pp. 25-30, 2011.

[DT04] W. Dally, B. Towles. *Principles and practices of interconnect networks*. Morgan Kaufmann Publishers, 2004.

[DTK01] C. Duan, A. Tirumala, and S. P. Khatri. *Analysis and avoidance of crosstalk in on-chip buses*. Proceedings of High Performance Interconnects (HOTI), 2001.

[Dua93] J. Duato. *A new theory of deadlock-free adaptive routing in wormhole networks*. IEEE Transactions on Parallel and Distribuited Systems, vol. 4, no. 12, pp. 1320–1331, 1993.

[DZK08] C. Duan, C. Zhu, S. P. Khatri. *Forbidden transition free crosstalk avoidance CODEC design*. Proceedings of Design Automation Conference (DAC), pp. 986-991, 2008.

[EAR10] A. Ejlali, B. M. Al-Hashimi, P. Rosinger, S. G. Miremadi, and L. Benini. *Performability/energy tradeoff in error-control schemes for on-chip networks*. IEEE Transactions on Very Large Scale Integrated System, vol. 18, no. 1, pp. 1-14, 2010.

[EK08] P.G. Emma and E. Kursur et al. *Is 3D chip technology the next growth engine for performance improvement?* IBM Journal of Research and Development, Vol. 52, No. 6, 2008.

[FDC09] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, D. Blaauw. *A highly resilient routing algorithm for fault-tolerant NoCs*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 21-26, 2009.

[FDG12] D. Fick, R. G. Dreslinski, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, D. Blaauw. *Centip3De: A 3930 DMIPS/W Configurable Near-Threshold 3D Stacked System with 64 ARM Cortex-M3 Cores*. Proceedings of, IEEE International Solid-State Circuits Conference (ISSCC), pp. 20-22, 2012.

[FKC06] A. P. Frantz, F. L. Kastensmidt, L. Carro, E. F. Cota. *Dependable Network-on-Chip Router Able to Simultaneously Tolerate Soft Errors and Crosstalk*.Proceedings of International Test Conference (ITC), pp. 1-9, 2006.

[FP09] S. Feero and P. P. Pande. *Networks-on-Chip in a Three-Dimensional Environment: A Performance Evaluation*. IEEE Transactions on Computers, vol. 58, no. 1, pp. 32-45, 2009.

[GBB08] A. Ganguly, P. P. Pande, B. Belzer, C. Grecu. *Design of Low Power & Reliable Networks on Chip Through Joint Crosstalk Avoidance and Multiple Error Correction Coding*. Journal of Electronic Testing: Theory and Applications (JETTA), vol. 24, no. 1-3, pp. 67-81, 2008.

[GIS06] C. Grecu, A. Ivanov, R. Saleh, E. S. Sogomonyan, P. P. Pande. *On-line Fault Detection and Location for NoC Interconnects*. Proceedings of International On-Line Test Symposium (IOLTS), pp. 145-150, 2006.

[GIS06a] C. Grecu, A. Ivanov, R. Saleh, P. P. Pande. *NoC Interconnect Yield Improvement Using Crosspoint Redundancy*. Proceeding of Symposyum on Defect and Fault Tolerance (DFT), pp. 457-465, 2006.

[GIS07] C. Grecu, A. Ivanov, R. A. Saleh, P. P. Pande. *Testing Network-on-Chip Communication Fabrics*. IEEE Transactions on CAD of Integrated Circuits and Systems vol. 26, no. 12, pp. 2201-2214, 2007.

[GM82] P. Goel and M.T. McMahon. *Electronic Chip-in-Place Test*. Proceedings of International Test Conference (ITC), pp. 83-90, 1982.

[GPI06] C. Grecu, P. P. Pande, A. Ivanov, R. Saleh. *BIST for Network-on-Chip Interconnect Infrastructures*. Proceedings of VLSI Test Symposium (VTS), pp. 30-35, 2006.

[GRL] Gaisler Aeroflex SoC Library. http://www.gaisler.com

[GWP09] M. Grange, R. Weerasekera, D. Pamunuwa, H. Tenhunen. *Examination of Delay and Signal Integrity Metrics in Through Silicon Vias*. 3D Integration Workshop. DATE Conference, 2009.

[HAG10] M. Healy, K. Athikulwongse, R. Goel et al. *Design and Analysis of 3D-MAPS: A Many-Core 3D Processor with Stacked Memory*. Proceedings of the IEEE Custom Integrated Circuits Conference (CICC), pp. 1-4, 2010.

[HGR07] A. Hansson, K. Goossens, A. Radulescu. *Analysis of Message-Dependent Deadlock in Network-Based Systems on Chip*. VLSI Design – Special issue on Networks-on-Chip, Hindawi Publishing, 2007.

[HHC10] C. Hsieh, T.T. Hwang, M.T. Chang, H.S. Tsai, C.M. Tseng, and H.-C. Li. *TSV redundancy: Architecture and design issues in 3D IC*. Proceedings of Design Automation and Test in Europe (DATE) Conference, pp. 166-171, 2010.

[HHH10] H.-Y. Huang, U.-S. Huang, C.-L. Hsu. *Built-in self-test/repair scheme for TSV-based three-dimensional integrated circuits*. Proceedings of Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 56-59, 2010.

[HJN08] D. Henry, F. Jacquet, M. Neyret, X. Baillin, T. Enot, V. Lapras, C. Brunet-Manquat, J. Charbonnier, B. Aventurier, N. Sillon. *Through silicon vias technology for CMOS image sensors packaging*. Proceedings of Electronic Components and Technology Conference (ECTC), pp.556-562, 2008.

[HLC11] Yu-Jen Huang, Jin-Fu Li, Ji-Jan Chen, Ding-Ming Kwai, Yung-Fa Chou, Cheng-Wen Wu. *A built-in self-test scheme for the post-bond test of TSVs in 3D ICs*. Proceedings of VLSI Test Symposium (VTS), pp.20-25, 2011.

[HM04] J. Hu, R. Marculescu. *Dyad: smart routing for networks-on-chip*. In Proceedings of Design Automation Conference, pp. 260 – 263, 2004.

[HRF11] C. Hernández, A. Roca, J. Flich, F. Silla, J. Duato. *Fault-Tolerant Vertical Link Design for Effective 3D Stacking*. IEEE Computer Architecture Letters, vol. 10, no. 2, pp. 41-44, 2011.

[HS00] R. Hegde and N. R. Shanbhag. *Toward achieving energy efficiency in presence of deep submicron noise*. IEEE Transactions on VLSI Systems, vol. 8, no. 4, pp. 379–391, 2000.

[HS04] Ching-Tien Ho and Larry Stockmeyer. A new approach to fault-tolerant wormhole routing for mesh-connected parallel computers. *IEEE Transactions on Computers*, vol. 53, no. 4, pp. 427–439, 2004.

[HSS09] C. Hernandez, F. Silla, V. Santonja, and J. Duato. A new mechanism to deal with process variability in NoC links. Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (ISPDP), pp. 1-11, 2009.

[ITR09] International Semiconductor Technology Roadmap (ITRS): Interconnects, 2009.

[JLV05] A. Jantsch, R. Lauter, A. Vitkowski. *Power analysis of link level and end-to-end data protection in networks on chip*. Proceedings of International Symposium on Circuits and Systems(ISCAS), pp. 1770- 1773, 2005.

[Jut04] A. Jutman. At-speed on-chip diagnosis of board-level interconnect faults. Proceedings of European Test Symposium (ETS), pp. 2- 7, 2004.

[JXE12] Li Jiang, Q. Xu, B. Eklow: On effective TSV repair for 3D-stacked ICs. Proceedings of Design Automation and Test in Europe Conference (DATE), pp. 793-798, 2012.

[JY89] N. Jarwala and C.W. Yau. *A New Framework for Analyzing Test Generation and Diagnosis Algorithms for Board Interconnects*. Proceedings of International Test Conference (ITC) pp. 63-70, 1989.

[JYX10] Li Jiang, Rong Ye, and Qiang Xu. *Yield enhancement for 3D-stacked memory by redundancy sharing across dies*. Proceedings of the International Conference on Computer-Aided Design (ICCAD), pp. 230-234, 2010.

[Kau74] W.H. Kautz. T*esting* of Faults in Wiring *Interconnects*. IEEE Transanctions on Computers, vol. 23, no. 4, pp. 358-363, 1974.

[KBS00] K. Kim, K. Baek, N. Shanbhag, C. Liu, S. Kang. *Coupling-driven signal encoding scheme for low-power interface design*. Proceedings of Internationa Conferece on Computer-Aided Design (ICCAD), pp. 318-321, 2000.

[KCH10] U. Kang, H.-J. Chung, S. Heo et al. *8 Gb 3-D DDR3 DRAM Using Through-Silicon-Via Technology*. IEEE Journal of Solid State Circuits vol. 45, no. 1, pp. 111-119, 2010.

[KM07] I. Koren and C. Mani Krishna. *Fault Tolerant Systems*. Morgan-Kaufman Publishers, 2007.

[KNG09] E. Kolonis, M. Nicolaidis, D. Gizopoulos, M. Psarakis, J. H. Collet, P. Zajac. *Enhanced self-configurability and yield in multicore grids*. Proceedings of International On-Line Test Symposium (IOLTS), pp. 75-80, 2009.

[KNM04] C. Kretzschmer, K. Nieuwland, D. Muller. *Why transition coding for power minimization of on-chip buses does not work*. Proceedings of Conference on Design, Automation and Test in Europe (DATE), pp. 512 – 517, 2004.

[KNM04] C. Kretzschmer, K. Nieuwland, D. Muller. *Why transition coding for power minimization of on-chip buses does not work*. Proceedings of the Conference on Design, Automation and Test in Europe, 2004.

[KXM09] A.P. Karmarkar, X. Xiaopeng Xu and V. Moroz. *Performanace and reliability analysis of 3D-integration structures employing Through Silicon Via (TSV)*. Proccedings of International Reliability Physics Symposium (IRPS), pp. 682 – 687, 2009.

[LAB08] I. Loi, F. Angiolini, and L. Benini. *Developing mesochronous synchronizers to enable 3D NoCs*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 1414-1419, 2008.

[LC04] S. Lin, D.J. Costello. *Error Control Coding: Fundamentals and Applications*. Second edition, Prentice Hall: Englewood Cliffs, NJ, 2004.

[LC09] H. H. S. Lee and K. Chakrabarty. *Test challenges for 3D integrated circuits*. IEEE Design & Test of Computers, vol. 26, pp. 26 - 35, September / October 2009.

[LCD09] X. Liu, O. Chen, P. Dixit, et al. *Failure mechanisms and optimum design for electroplated copper Through-Silicon Vias (TSV)*. Proceeding of Electronic Components and Technology Conference, pp. 624 – 629, 2009.

[LHZ06] Z. Li, X. Hong, Q. Zhou, S. Zeng, J. Bian, H. Yang, V. Pitchumani, and C.-K. Cheng. *Integrating dynamic thermal via planning with 3D floorplanning algorithm*. Proceedings of the International Symposium on Physical design (ISPD) pp. 178-185, 2006.

[LLD10] L. Jiang, Y. Liu, L. Duan, Y. Xie, Q. Xu. *Modeling TSV Open Defects in 3D-Stacked DRAM*. Proceedings of the IEEE International Test Conference (ITC), pp. 1-9, 2010.

[LLI] Low Latency Interface. MiPi Alliance http://www.mipi.org/lli

[LML08] I. Loi, S. Mitra, T. H. Lee, S. Fujita, and L. Benini. *A low-overhead fault tolerance scheme for TSV-based 3D network on chip links*. Proceedings of the International Conference on Computer-Aided Design (ICCAD), pp. 598-602, 2008.

[Loh08] G. H. Loh. 3D-*Stacked Memory Architectures for Multi-Core Processors*. Proceedings of 35th ACM International Symposium on Computer Architecture, pp. 453-464, 2008.

[LSL11] C. Liu, T. Song, S. K. Lim. *Signal Integrity Analysis and Optimization for 3D ICs*. Proceeding of IEEE International Symposium on Quality Electronic Design, pp.1-8, 2011.

[LSL11a] C. Liu, T. Song, et al. *Full-Chip TSV-to-TSV Coupling Analysis and Optimization in 3D IC*. Proceedings of Design Automation Conference (DAC), pp. 783-788, 2011.

[Lu03] J.-Q. Lu, K.W. Lee, Y. Kwon, G. Rajagopalan, J. McMahon, B. Altemus, M. Gupta, E. Eisenbraun, B. Xu, A. Jindal, R.P. Kraft, J.F. McDonald, J. Castracane, T.S. Cale, A. Kaloyeros, and R.J. Gutmann. *Processing of Inter-Wafer Vertical Interconnects in 3D ICs*. Proceedings of Advanced Metallization Conference (AMC), pp. 45-51, 2003.

[Mar10] E. J. Marinissen. *Testing TSV-based three-dimensional stacked ICs*. Proceedings of Design, Automation and Test in Europe (DATE) Conference, pp. 1689-1694, 2010.

[MCK12] E. J. Marinissen, C.-C. Chi, M. H. Konijnenburg, J. Verbree. A DfT Architecture for 3D-SICs Based on a Standardizable Die Wrapper. Journal of Electronic Testing vol. 28 no. 1, pp. 73-92, 2012.

[MTV05] S. Murali, T. Theocharides, N. Vijaykrishnan, M.J. Irwin, L. Benini, G. De Micheli. *Analysis of Error Recovery Schemes for Networks on Chips*. IEEE Design & Test of Computers. No. 22, Vol.5, September 2005.

[MZ09] E. J. Marinissen, Y. Zorian. *Testing 3D chips containing through-silicon vias*. Proceedings of International Test Conference (ITC), pp. 1-11, 2009.

[NCG11] B. Noia, K. Chakrabarty, S. K. Goel, E. J. Marinissen, J. Verbree. *Test-Architecture Optimization and Test Scheduling for TSV-Based 3-D Stacked ICs*. IEEE Transactions on CAD of Integrated Circuits and Systems vol. 30, no. 11, pp. 1705-1718, 2011.

[NGC10] B. Noia, S.K. Goel, K. Chakrabarty, E.J. Marinissen, J. Verbree. *Test-architecture optimization for TSV-based 3D stacked ICs* . Proceedings of European Test Symposium, ETS, pp. 24-29, 2010.

[Nic05] M. Nicolaidis. Design for soft-error mitigation. *IEEE Transactions on Device and Materials*, vol. 5, no. 3, pp. 405-418, 2005.

[Pas09] S. Pasricha. *Exploring serial vertical interconnects for 3D ICs*. Proceedings of the 46th Annual Design Automation Conference (DAC), pp. 581-586, 2009.

[Pat06] R. Patti. *Three-Dimensional Integrated Circuits and the Future of System-on-Chip Designs*. Proceedings of the IEEE, vol. 84, no. 6, pp. 1214 – 1224, 2006.

[Pat11] R. Patti. *Advances in 3D integrated circuits*. Proceedings of International Symposium on Physical Design, ISPD, pp. 79-80, 2011.

[PCF07] P. Leduc, F. de Crecy, M. Fayolle, B. Charlet, T. Enot, M. Zussy, et al. *Challenges for 3D IC integration: bonding quality and thermal management*. Proceedings of the IEEE International Interconnect Technology Conference (IITC), pp. 210–212, 2007.

[PCZ01] R. Pendurkar, A. Chatterjee, Y. Zorian. *Switching activity generation with automated BIST synthesis for performance testing of interconnects*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), vol. 20, no. 9, pp.1143-1158, 2001.

[PCZ01] R. Pendurkar, A. Chatterjee, Y. Zorian. Switching activity generation with automated BIST synthesis for performance testing of interconnects. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), vol. 20, no. 9, pp.1143-1158, 2001.

[PDA08] A. K. Palit, K. K. Duganapalli, W. Anheier. *Crosstalk fault modeling in defective pair of interconnects*. Integration, VLSI Journal vol. 41, no. 1, pp. 27-37, 2008.

[PED08] D. Park, S. Eachempati, R. Das, A. K. Mishra, Y. Xie, N. Vijaykrishnan, C. R. Das. *MIRA: A Multi-layered On-Chip Interconnect Router Architecture*. Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA), pp. 251-261, 2008

[PF07] V. Pavlidis, E. Friedman. *3-D topologies for networks-on-chip*. IEEE Transactions on Very Large Scale Integrated Systems, vol. 15, no. 10, pp. 1081-1090, 2007.

[PF09] V. F. Pavlidis, E. G. Friedman. *Three-Dimensional Integrated Circuit Design*. Morgan Kaufmann Publishers, ISBN: 978-0-12-374343-5, 2009.

[PG07] I. M. Panades, A. Greiner. Bi-Synchronous FIFO for Synchronous Circuit Communication Well Suited for Network-on-Chip in GALS Architectures. Proceedings of Network-on-Chip Symposium (NOCS), pp. 83-94, 2007.

[PKC10] M. Palesi, S. Kumar, and V. Catania. Leveraging partially faulty links usage for enhancing yield and performance in networks-on-chip. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, no. 3, pp. 426-440, March 2010.

[PLB04] M. Pirretti, G. M. Link, R. R. Brooks, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin. Fault tolerant algorithms for network-on-chip interconnect. In Proceedings of IEEE Computer society Annual Symposium on VLSI, pp, 46 – 51, 2004.

[PNK06] D. Park, C. Nicopoulos, J. Kim, N. Vijaykrishnan, C. R. Das. Exploring fault-tolerant network-on-chip architectures. Proceedings of International Conference on Dependable Systems and Networks (DSN), pp. 93-104, 2006.

[PZ11] S. Pasricha, Y. Zou. *A low overhead fault tolerant routing scheme for 3D Networks-on-Chip*. Proceedings of ISQED, pp. 204-211, 2011.

[PZG06] P. P. Pande, H. Zhu, A. Ganguly, C. Grecu. *Energy reduction through crosstalk avoidance coding in NoC paradigm*. Proceedings of EUROMICRO Conference on Digital System Design *(DSD)*, pp. 689-695, 2006.

[QLD09] Y. Qian, Z. Lu, and W. Dou. From 2D to 3D NoCs: a case study on worst-case communication performance. Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 555-562, 2009.

[RAA10] C. Rusu, L. Anghel, D. Avresky. *Adaptive inter-layer message routing in 3D networks-on-chip*. Microprocessors and Microsystems - Embedded Hardware Design, Vol. 35, no. 7, pp. 613-631.

[RAM07] D. Rossi, P. Angelini, C. Metra. *Configurable Error Control Scheme for NoC Signal Integrity*. Proceedings of 13th IEEE International On-Line Testing Symposium, July 2007.

[RFR10] S. Rodrigo, J. Flich, A. Roca, S. Medardoni, D. Bertozzi, J. Camacho, F. Silla, J. Duato. *Addressing Manufacturing Challenges with Cost-Efficient Fault Tolerant Routing*. Proceedings of the ACM/IEEE International Symposium on Networks-on-Chip (NOCS), pp. 25-32, 2010.

[SAS04] S. R. Sridhara, A. Ahmed, N. R. Shanbhag. *Area and energy-efficient crosstalk avoidance codes for on-chip buses*. Proceedings of Interactional Conference on Computer Design (ICCD), pp. 12-17, 2004.

[SC11] S. Shamshiri, K.-T. Tim Cheng. *Modeling yield, cost, and quality of a spare-enhanced multi-core chip. IEEE Transactions on Computers, Special issue on* Concurrent On-Line Testing and Error/Fault Resilience of Digital Systems, 2011.

[SCL] SoCLiB SystemC Library. http://www.soclib.fr

[SD02] K. Sekar, S. Dey. *LI-BIST: a low-cost self-test scheme for SoC logic cores and interconnects.* Proceedings of VLSI Test Symposium (VTS), pp. 417- 422, 2002.

[Sin11] E. Singh. *Exploiting rotational symmetries for improved stacked yields in W2W 3D-SICs*.Ptoceedings of VLSI Test Symposium (VTS), pp. 32-37, 2011.

[SMB09] C. Seiculescu, S. Murali, L. Benini, G. De Micheli. *SunFloor 3D: a tool for networks on chip topology synthesis for 3D systems on chips*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 9-14, 2009.

[SN96] K.L. Shepard, V. Narayanan. *Noise in deep submicron digital design*. In Proceedings of IEEE/ACM International Conference on CAD, November, 1996.

[SS05] S. Sridhara, N. Shanbhag. *Coding for system-on-chip networks: A unified framework*. IEEE Transactions on VLSI, 13(2):665-667, June 2005.

[TCL10] T. Frank, C. Chappaz, et al. *Reliability approach of high density Through Silicon Via (TSV)*. Proceedings of 12th Electronics Packaging Technology Conference (EPTC), pp. 312-324, 2010.

[TH11] M. Taouil, S. Hamdioui. *Layer Redundancy Based Yield Improvement for 3D Wafer-to-Wafer Stacked Memories*. Proceedings of European Test Symposium (ETS) pp. 45-50, 2011.

[THV10] M. Taouil, S. Hamdioui, J. Verbree, E. J. Marinissen. *On maximizing the compound yield for 3D Wafer-to-Wafer stacked ICs*. Proceedings of International Test Conference (ITC), pp. 183-192, 2010.

[TLP07] T. Lehtonen, P. Liljeberg, and J. Plosila. *Online reconfigurable self-timed links for fault tolerant NoC*. VLSI Design, vol. 2007, pp. 1-13, 2007.

[TMS07] R. Tamhankar, S. Murali, S. Stergiou, A. Pullini, F. Angiolini, L. Benini, G. De Micheli. *Timing-Error-Tolerant Network-on-Chip Design Methodology*. IEEE Transactions on CAD of Integrated Circuits and Systems, vol. 26, no. 7, pp. 1297-1310, 2007.

[TVC10] Y. Thonnart, P. Vivet, F. Clermidy. *A fully-asynchronous low-power framework for GALS NoC integration*. Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 33-38, 2010.

[VDG03] B. Vermeulen, J. Dielissen, K. Goossens, C. Ciordas. *Bringing Communication Networks On Chip: Test and Verification Implications*. IEEE Communications Magazine, vol. 41, no 9, pp. 74-81, 2003.

[VG11] P. Vivet, V. Guérin. A *Three-Layers 3D-IC Stack including Wide-IO and 3D NoC (Network on Chip) – Practical Design Perspective*. Proceedings of 3D-Architectures for Semiconductor Integration and Packaging (3D-ASIP). 2011.

[VK01] B. Victor, K. Keutzer. Bus encoding to prevent crosstalk delay. Proceedings of I*nternational Conference on Computer-Aided Design* (ICCAD), pp. 57-63, 2001.

[VMM09] D. Velenis, M. Stucchi, E.J. Marinissen, B. Swinnen, and E. Beyne. *Impact of 3D design choices on manufacturing cost*. Proceedings of the IEEE International Conference on 3D System Integration (3DIC), pp. 1-5, 2009.

[VSN10] A. Vitkovskiy, V. Soteriou, C. Nicopoulos. *A fine-grained link-level fault-tolerant mechanism for networks-on-chip*. Proceedings of the 28th IEEE International Conference on Computer Design (ICCD), pp.447-454, 2010.

[Wag87] P.T. Wagner. *Interconnect Testing* with Boundary Scan. Proceedings of International Test Conference (ITC), pp. 52-57, 1987.

[YA10] Q. Yu and P. Ampadu. *Transient and Permanent Error Co-management Method for Reliable Networks-on-Chip*. Proceedings of the 4th ACM/IEEE International Symposium on Networks-on-Chip (NOCS), pp. 145-154, 2010.

[YL08] S. Yan, B. Lin. *Design of application-specific 3D Networks-on-Chip architectures*. Proceedings of International Conference on Computer Design (ICCD), pp.142-149, 2008.

[ZLS02] Y. Zhang, J. Lach, K. Skadron, M. Stan. *Odd/even bus invert with two-phase transfer for buses with coupling*. Proceedings of International Symposium on Low Power Electronics and Design (ISLPED), pp. 80-83, 2002.

[ZGT08] Z. Zhang, A. Greiner, S. Taktak. *A reconfigurable routing algorithm for a fault-tolerant 2D-Mesh Network-on-Chip*. Proceedings of Design Automation Conference (DAC), pp. 441-446, 2008.

[ZGB10] Z. Zhang, A. Greiner, M. Benabdenbi. *Fully distributed initialization procedure for a 2D-Mesh NoC, including off-line BIST and partial deactivation of faulty components*. Proceedings of International On-Line Test Symposium (IOLTS), pp. 194-196, 2010.

[ZGS08] C. Zhu, Z. Gu, Li Shang, R. P. Dick, R. Joseph. *Three-Dimensional Chip-Multiprocessor Run-Time Thermal Management*. IEEE Transactions on CAD of Integrated Circuits and Systems vol. 27 no. 8, pp. 1479-1492, 2008.

[ZKA11] Y. Zhao, S. Khursheed, B. M. Al-Hashimi. *Cost-Effective TSV Grouping for Yield Improvement of 3D-ICs*. Proceedings of Asian Test Symposium (ATS), pp. 201-206, 2011.

[ZRG11] Z. Zhang, D. Refauvelet, A. Greiner, M. Benabdenbi, F. Pêcheux. *Localization of damaged resources in NoC based shared-memory MP2SOC, using a Distributed Cooperative Configuration Infrastructure*. Proceedings of VLSI Test Symposium (VTS), pp. 229-234, 2011.

Publications

[PAB] Vladimir Pasca, Lorena Anghel, Mounir Benabdenbi. KAF-based TSV Interconnect Testing. Journal of Electronic Testing: Theory and Applications. accepted for publication

[NPA12] Michael Nicolaidis, Vladimir Pasca, Lorena Anghel. Through-Silicon-Via Built-In Self-Repair for Aggressive 3D Integration. Proceedings of International On-Line Test Symposium (IOLTS 2012).

[PRA12] Vladimir Pasca, Saif-Ur Rehman, Lorena Anghel, Mounir Benabdenbi. Efficient Link-level Error Resilience in 3D NoCs. Proceedings of International Symposyum on Designs and Diagnosis of Electronic Circuits and Systems (DDECS 2012), pp. 135-140, 2012.

[PAN12] Vladimir Pasca, Lorena Anghel, Michael Nicolaidis, Mounir Benabdenbi. CSL: Configurable Fault Tolerant Serial Links for Inter-die Communication in 3D Systems. Journal of Electronic Testing: Theory and Applications, vol. 28, no.1, pp. 137-150, 2012

[NPA11] Michael Nicolaidis, Vladimir Pasca, Lorena Anghel. I-BIRAS: Interconnect Built-In Self-Repair and Adaptive Serialization for Inter-Die Communication in 3D Integrated Systems. Proceedings of European Test Symposium, ETS 2011, Trondheim, Norway.

[PAB11] Vladimir Pasca, Lorena Anghel, Mounir Benabdenbi. Configurable TSV Interconnect Built-In Self-Test. Proceedings of Latin America Test Symposium, LATW 2011, pp. 115-120, 27-30 March, Brazil.

[NAP10] Michael Nicolaidis, Lorena Anghel, Vladimir Pasca. I-BIRAS: Interconnect Built-In Self-Repair and Adaptive-Serialization. Proceedings of International Test Conference Workshop on Test of 3D Stacked Systems, 4-5 November 2010, Austin, USA.

[PAB10b] Vladimir Pasca, Lorena Anghel, Mounir Benabdenbi. Fault Resilient Intra-die and Inter-die Communication in 3D Integrated Systems. Proceedings of PhD Research in Microelectronics and Electronics Conference, PRIME 2010, 18-21 July, Berlin, Germany.

[NPA10] Michael Nicolaidis, Vladimir Pasca, Lorena Anghel. Interconnect Built-In Self-Repair and Adaptive-Serialization (I-BIRAS) for 3D Integrated Systems. Proceedings of International On-Line Test Symposium, IOLTS 2010, pp. 115-120, ISBN: 978-1-4244-7721-0, 3-7 July, Corfu, Greece.

[PAR10d] Vladimir Pasca, Lorena Anghel, Claudia Rusu, Mounir Benabdenbi. Configurable Serial Fault-Tolerant Link for Communication in 3D Integrated Systems. Proceedings of International On-Line Test Symposium, IOLTS 2010, page 115-120, ISBN: 978-1-4244-7721-0, 3-7 July, Corfu, Greece.

[PAB10a] Vladimir Pasca, Lorena Anghel, Mounir Benabdenbi. Fault Tolerant Communication in 3D Integrated Systems. Proceedings of DSN Workshop on Dependable Systems and Networks, WDSN 2010, pp. 131-135, ISBN: 978-1-4244-7728-9, 28 June, Chicago, USA.

[PAR10c] Vladimir Pasca, Lorena Anghel, Claudia Rusu, Mounir Benadenbi. Non-regular 3D mesh Networks-on-Chip. Prooceedings of DAC Workshop on Diagnostic Services in Network-on-Chips, DSNoC 2010, 13 June, Anaheim, USA.

[PAR10b] Vladimir Pasca, Lorena Anghel, Claudia Rusu, Mounir Benabdenbi. Configurable Fault-Tolerant Link for Inter-die Communication in 3D on-Chip Networks. Proceedings of European Test Symposium ETS 2010, pp. 258, ISBN: 978-1-4244-5833-2, 24-28 May, Prague, Czech Republic.

[PAR10a] Vladimir Pasca, Lorena Anghel, Claudia Rusu, Riccardo Locatelli, Marcello Coppola. Error Resilience of Inter-Die and Intra-Die Communication with 3D Spidergon STNoC. Proceedings of Design Automation and Test in Europe Conference DATE 2010, pp. 275-278, ISBN: 978-3-9810801-6-2, 8-12 March Dresden, Germany.

APPENDIX

The error resilience protection strategies presented in Chapter 4 have been implemented for the seven-port router used in 3D mesh topologies. In this appendix, the router and 3D mesh NoC architectures are presented in detail. Moreover, a simple design flow for 3D mesh network is presented along with its support for error resilient strategies.

A. Fully synchronous 3D mesh NoC

In 3D MP2SoCs (Massively-Parallel Multi-Processor Systems-on-Chip), 3D mesh NoC can be used as the global interconnect fabric. In the 3D mesh topology, all nodes and horizontal links are intra-die components and the vertical connections correspond to TSV-based inter-die links. The seven-port router with its interfaces is the main component. In this appendix, a 3D mesh NoC implementation and design flow is presented.

A.1 Packet format

In the 3D mesh NoCs, packets are router from source to destination along the Z, Y and X dimensions. For each load / store transaction, the network interface *NI* translates the transaction message destination address into the destination coordinates X_{DEST}, Y_{DEST}, Z_{DEST}. The packet is then built and transmitted through the network *flit-by-flit*. The header flit is created by the *NI* and it contains the destination node coordinates (X_{DEST},Y_{DEST},Z_{DEST}). These coordinates, along with the local coordinates, are used by routing nodes to forward packets through the network. Each *n*-bits flit has *n-1* data flits and an *End-of-Packet* (EoP) marker, which is active for the last flit of the packet. In Figure A-1, the 3D mesh NoC topology and the packet structure are represented.

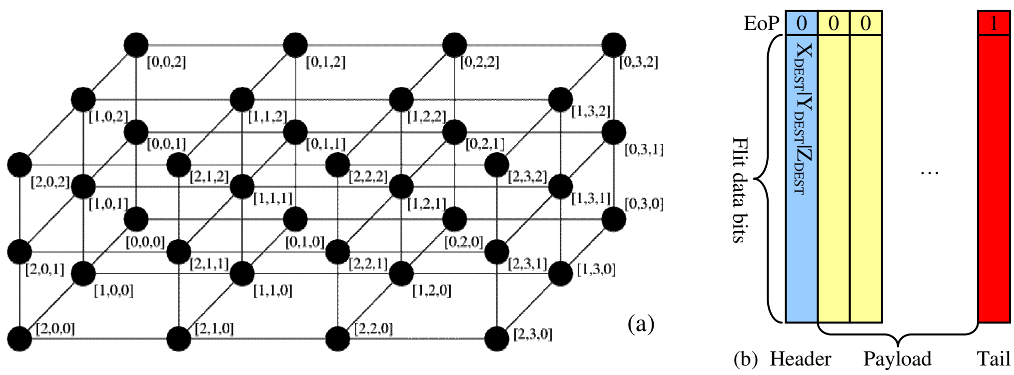


Figure A-1 3D 3×4×3 mesh topology (a) and the packet format for the 3D fully-synchronous 3D NoC (b)

The network processes packets and routes them from source to destination using the *ZYX* routing algorithm. Each router reads the destination coordinates from the header flit and forwards the packet the direction indicated for by the routing algorithm. Once the router output port is allocated to a packet, it cannot be de-allocated until the *EoP* marker is valid. Details about the router architecture are presented in the following.

A.2 Router architecture[2]

The 3D NoC is fully-synchronous, as all NoC components (i.e. links and routers) are in the same clock domain. In a 3D mesh, packets are routed along the X, Y and Z directions. Each router has up to seven ports: one for communication with the local IP (i.e. LOCAL), four (i.e. *NORTH*, *SOUTH*, *EAST*, and *WEST*) for intra-die communications, and two (i.e. *UP*, *DOWN*) for inter-die communication. Each of the input / output ports is buffered, in order to maximize network performance. In Figure A-2, the router architecture is given.

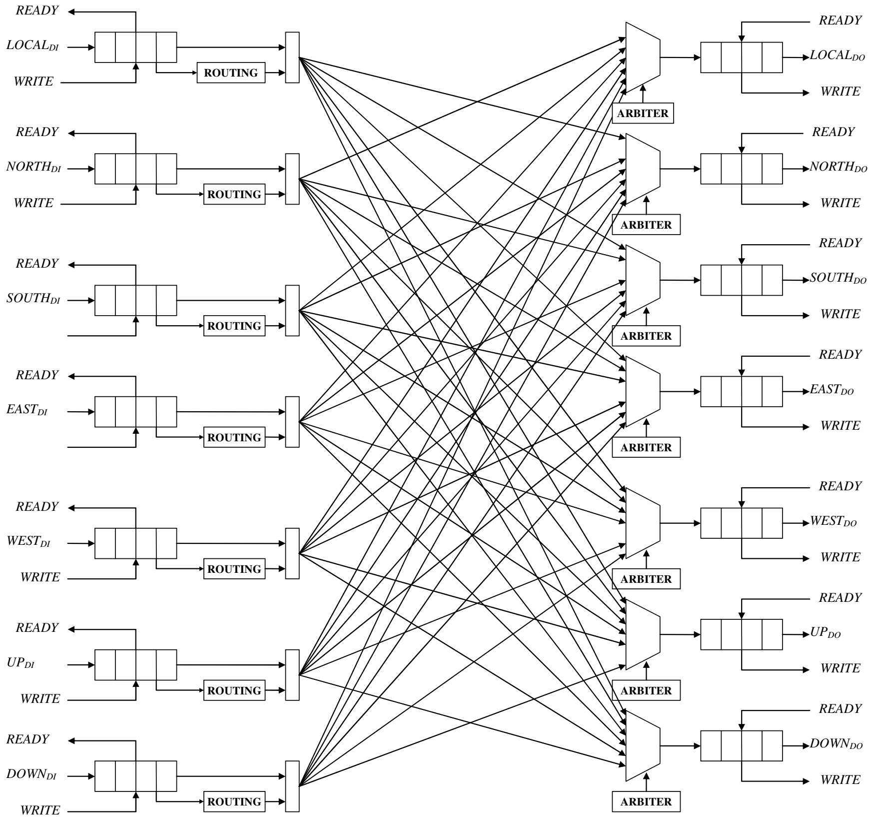


Figure A-2 Seven-port router micro-architecture

Routers exchange data using the X_{ON}/X_{OFF} flow control. Data is sent from the upstream router to the downstream router input FIFO when there are enough available positions in the downstream FIFO. Since the link delay is one clock cycle, downstream FIFOs accept data when there is at least one available position. This flow control mechanism is implemented using two control signals: the upstream-to-downstream *write* signal and the downstream-to-upstream *ready* signal.

Incoming flits are stored in input FIFOs, which are handled by *FSM*s. The input *FSM* has two states: *idle* and *flush*. In the *idle* state the input FIFO read signal is '0'. The *FSM* makes the transitions to flush when the input buffer is not empty (i.e. $read_{ok}$*='1'*) and the first flit in the FIFO is not a tail flit (i.e. *EoP='0'*). In the *flush* state flits are read input FIFO and written in the output FIFO designated by the *routing* module. The routing logic interprets the destination address and indicates on which output port the packet must be forwarded.

---

[2] The router architecture is based on the synchronous five-port DSPIN router used in SoCLib

Output buffers are also managed by FSMs. For each output port there is an arbiter that manages requests from input ports. If input *I* requests output *O* and this request is granted then, while the output FIFO *O* is not full, data is transferred from the input FIFO to the output FIFO. After the last flit is transferred, the arbiter will grant a new request using the *Round-Robin* algorithm. Let's consider a four-flit packet that traverses the router from input *I* to output *O*. The output FSM has *12* states that correspond to the cases when the output is granted to one of the six input ports.

To illustrate the router functionality, les us consider a packet with four flits *A*, *B*, *C*, and *D*. The header flit A contains the destination coordinates, flits B and C are payload flits, and flit D is the tail flit. In Figure A-3, the time diagram of router traversal is represented for the four-flit packet.

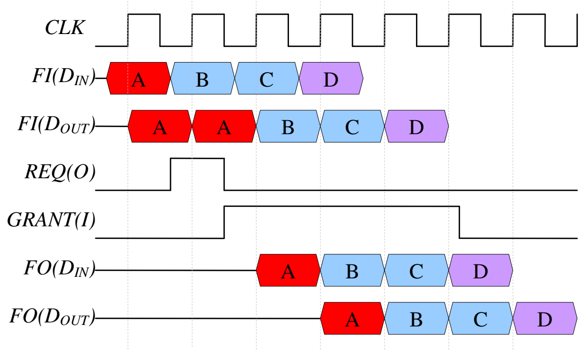


Figure A-3 Time diagram of contention-free router traversal from input I to output O

In the first cycle the header flit A is written in port *I*'s input FIFO *FI*. Then, since data exists in the FIFO the routing algorithm function determines the output port O on which the packet must be forwarded and activates the request signal *REQ(O)*. The arbitration logic of output *O* grants this request (i.e. *GRANT(I)*) in the third cycle. Note that if there are more packets competing for the same output *O* then the grant signal comes later. Once this grant is received data is transferred from the input to the output FIFO. This transfer is stalled if the output FIFO is full, but it is resumed when there are available positions. In the fourth cycle the header flit is read from the output FIFO by the downstream router. After the tail flit is written in the output FIFO the grant signal is deactivated and a new arbitration process starts. Because routing and arbitration is done in a single cycle, data waits at the input FIFO for only one cycle.

The seven-port router is the main component of the 3D mesh NoC, which consists of many interconnected routers. The router can be implemented as a *RTL* module with generic local coordinates X_{LOCAL}, Y_{LOCAL}, *and* Z_{LOCAL}, data size, and buffer size. For this router, a 3D NoC instantiation tools has been developed. This design flow is presented in the following section.

A.3 3D NoC Design Flow

The seven-port router presented above can be integrated in an automatic 3D NoC design flow. The NoC designer sets the network parameters (i.e. 3D mesh X, Y, Z sizes, flit size, and minimum buffer size) and the RTL code of the 3D NoC is automatically generated. For the *4×4×4* 3D mesh, the tool generates the *noc_4x4x4* component, which has *64* input / output communication interfaces and *64* instantiations of the *fsr_router* component. In the section, details about the 3D NoC simulation platform and the support for error resilience interfaces are presented.

A.4 Simulation platform

The design tool also has the capability to generate simulation platforms for the instantiated NoC. Thus, traffic injection and analysis models have been implemented. In the current version of the tool only random uniform traffic is supported. In the simulation platform, a traffic input/output module is connected to each NoC port. In Figure A-4, an example of a simulation platform for the *noc_xyz* module is presented.

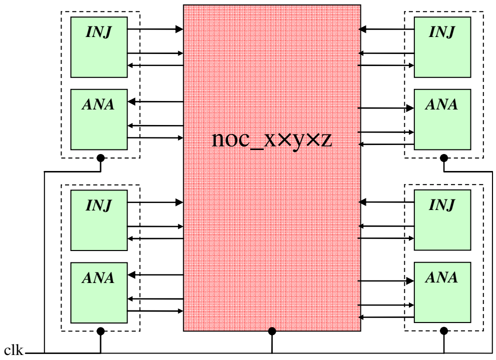


Figure A-4 Simulation platform for noc_xyz module

In the random uniform traffic model, packets are transmitted between random pair of nodes. The packet size can be fixed (i.e. the same packet length for all traffic modules) or random (i.e. the packet lengths may vary within a predetermined range l_{min} and l_{max}), but the payload contents is randomly generated. The header flit contains the X,Y,Z coordinates of the destination node an a timestamp that corresponds to the moment when the packet was created. During simulation, packets are analyzed by the traffic modules and the latency of each packet is printed on the output. The latency is determined from the timestamp field in the header flit and the current time when the tail flit is processed the *ANA* traffic sub-module.

The latencies of all packets can then be collected and different metrics such as average latency, latency distributions and throughput can be determined.

A.5 Support for error resilience

In order to support the error resilience schemes presented in Chapter 4, the 3D design flow must be extended in a number of ways. The interfaces for intra-die (i.e. *NORTH*, *SOUTH*, *EAST*, and *WEST*) and inter-die (i.e. UP and DOWN) communication are part of *link* modules, which are used for *router-to-router* connections, while the *LOCAL* interfaces are instantiated in the top module. For area / power assessments, the ERX uses an instantiation of the seven-port router with error resilient interfaces. In Figure A-5, an example of error resilient router is shown.

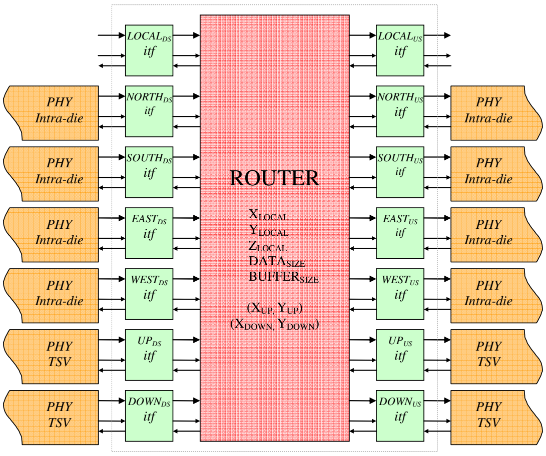


Figure A-5 Instantiated seven-port router with error resilience interfaces

In the modified flow, the upstream / downstream interfaces form *link* modules. The *noc_xyz* module, which is the instantiated 3D NoC, consists of *routers* and *links*. In Figure A-6, an example of two connected routers is shown.

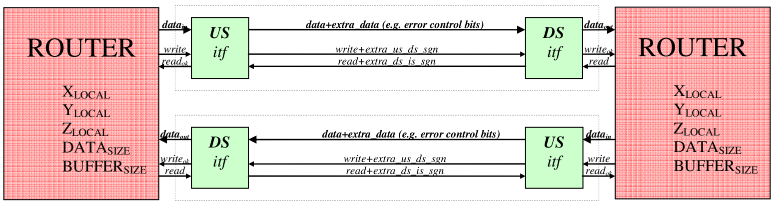


Figure A-6 Link instantiation beween two connected *router* ports using two *link* modules

In the error resilient framework, each *router* is modified such that the master-up / master-down coordinates are given as inputs. The signals that carry these values are determined by *ERX* and they are fixed during simulation. Each link that connects two routers consists of two interfaces: *upstream* and *downstream*. When the link does not implement any error resilience scheme, the interfaces simply forward the signals coming from routers.

The data link error resilience schemes are implemented at these interfaces. Extra wires are required for the error control bits of the coding scheme and the data/code wire spares. In some cases the communication between the upstream and downstream interfaces is more complex. Thus, additional *us-to-ds* and *ds-to-us* control signals (e.g. retransmission request for *ARQ*, D_{TRANS} for IBISnR/IBIRAS) must be added. The interfaces for the six inter-router ports implement the data link error resilience schemes. These modules are generic, allowing different configurations. In Figure A-7, the upstream and downstream interfaces with the optional retiming stages are shown.

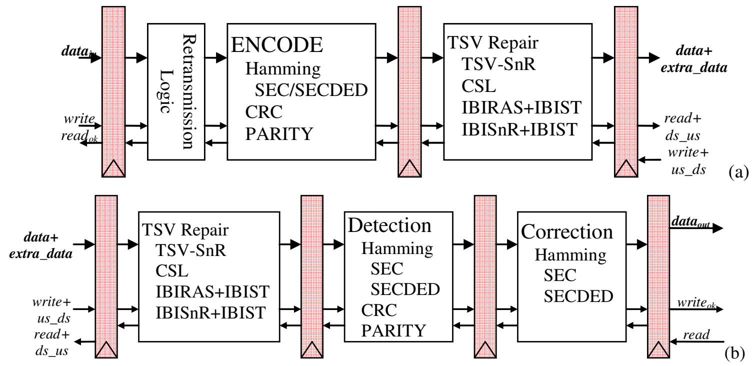


Figure A-7 Upstream (a) and downstream (b) interfaces with optional modules and register stages

Retiming stages are inserted in order to improve circuit timing. In the ERX area / power evaluation phase, these retiming stages are added such that the circuit meets the timing requirements. Initially, only the first retiming stages of the downstream interface is considered, as it is considered that encoding, *PHY* traversal, and decoding cannot be performed in a single clock cycle. If timing is not met, the retiming stage between the detection / correction modules is added. The stages between the *ENCODE / TSV-Repair* and *TSV-Repair / Detection* modules are added last. The *upstream* output registers are added if transmission on *PHY* is assumed to be exactly one cycle, while the *us* input and *ds* output are merged with the router FIFOs.

The *LOCAL* downstream interface ($LOCAL_{DS}$) of the router is connected to the local network interface (*NI*). In the *NL-FEC* error resilient configuration, the Hamming SEC encoding stage is implemented. At the *LOCAL* upstream interface error cumulated along the network are corrected. In the hardware implementation, this interface may have an optional bypass mechanism for the error correction module.

A.6 Fault injection

In order to perform simulations using the error resilient platform, errors must be injected in the NoC. The generated simulation platform enables more accurate measurements of network performance metric when error resilience schemes are implemented and errors are injected. In Figure A-8, the simulation platform for the error resilient NoC is presented.

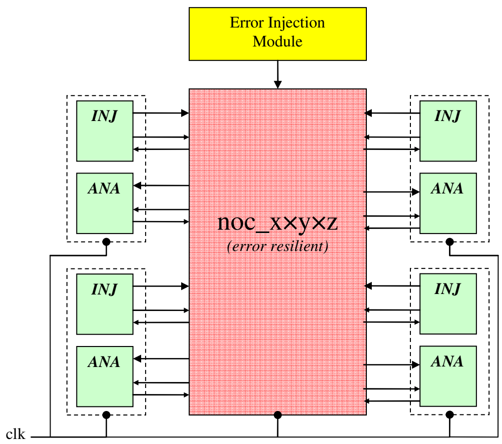


Figure A-8 Simulation platform for error resilient noc_xyz module with error injection for intra-/inter-die wires

For permanent TSV failures due to manufacturing defects, a fault indicator *FI* vector is generated for each inter-die link in the 3D NoC instantiation process. Using this vector, spare-/serialization-based repair is performed. In the TSV-FTR algorithm the master *UP* and *DOWN* of each node are determined by the *master node selection* algorithm. Since the master nodes do not change during system lifetime, the signals that carry their values in the *noc_xyz* module are initialized in the instantiation process. In the current version of the tool, there is no support for simulations with in-field TSV failures. Thus, *IBIRAS* techniques are used in a similar way as *CSL / TSV-SnR* (i.e. the fault indication vector is set in the *noc-xyz* instatiation).

The error resilient simulation platform has an extra module for injecting transient errors in the protected links of *noc_xyz*. Transients are injected intra-/inter-die links with a probability ε_{link}, which is determined by *ERX* using the ε_{TSV} and ε_{wire} parameters. The current version of the tool allows only single transients injection.

B. Spidergon-based 3D GRLIB MPSoC

The Gaisler IP library (GRLIB) [GR] is a highly configurable open-source collection of RTL-level IPs developed around the LEON3 microprocessor. In order to simplify the integration process, all the IP blocks are AMBA AHB/APB-compliant. GRLIB comes with a series of template designs that can be easily implemented using FPGA or ASIC design flows. In a 3D integration setting, the template design GRLIB design is extended with two addressable components (i.e. on-chip memories), which are connected to the AHB-bus through the Spidergon STNoC [CGM09] AXI-based interconnect fabric.

In this section, the impact of error resilience on the Spidergon STNoC-based system is assessed at system level. First, the Spidergon STNoC and the STNoC link, which ensures inter-die communication in the 3D SoC, are presented. The implementation of the extended GRLIB-based MPSoC is presented next, followed by the proposed error resilience strategy. Finally, the impact on system performance is assessed.

B.1 Spidergon STNoC

Spidergon STNoC is a high performance customizable on-chip communication platform that addresses heterogeneous, application specific requirements of multi-core SoCs (MPSoC). It allows fully customizable pseudo-regular or hierarchical topologies. As a programmable distributed hardware / software component, Spidergon STNoC offers a set of services to design advanced application features such as quality of service, security, exception handling. The building blocks of Spidergon STNoC are the network interfaces, routers and links. Each of these blocks implements specific levels in the ISO/OSI protocol layers. In stacked 3D integrated heterogeneous SoCs, the IP blocks of the system are distributed over several levels of the stack. The communication infrastructure (NoC) must connect such blocks and, the resulting communication structure contains both horizontal (intra-die) and vertical (inter-die) links (3D NoC). In Figure A-9, different 3D Spidergon STNoC topologies are represented.

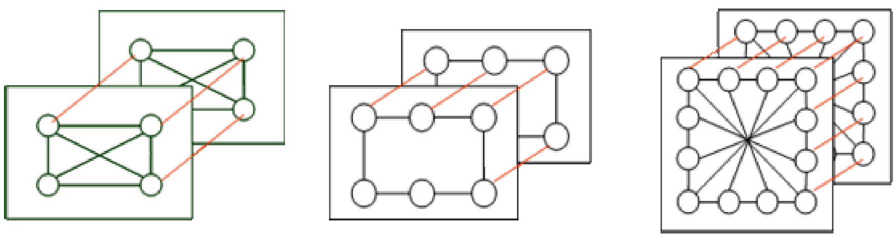


Figure A-9 Spidergon STNoC topologies

The Spidergon STNoC router implements both the network and the data link layers of the NoC protocol, offering Quality of Service (QoS) in terms of both latency and throughput. It is responsible for the flits transmission using wormhole routing protocols. The STNoC router is designed to support ST's proprietary Spidergon topology and it has five input / output buffered ports to the network interface (*NI*), left (*L*), right (*R*), across (*A*) and hierarchical (*H*). Depending on the application traffic requirements, links can be removed, which gives STNoC the key benefit of being able to support with one homogeneous component a wide spectrum of MPSoC interconnect topologies.

At the lowest level, the STNoC Physical link implements the physical layer of the NoC protocol. It is responsible for the connections between routers and between routers and network interfaces. In 3D MPSoCs, the STNoC link is used for inter-die communication. Details about the synchronous link are presented in the following.

B.2 Synchronous Spidergon STNoC link

There are several possible ways of implementing physical links: synchronous or asynchronous, with different flit sizes, etc. Of course, the choice of physical link technology involves trade-offs between many issues such as clock distribution over a wide silicon area, amount of on-chip wiring and the chip area required. In this respect, the decoupling of layers provided by the NoC paradigm is a major advantage, as changes to the physical layer can be subsequently made without affecting the packet transport and transaction layers. The Spidergon STNoC link represented in Figure A-10 implements the physical level of the communication structure using intra-die or inter-die interconnects.

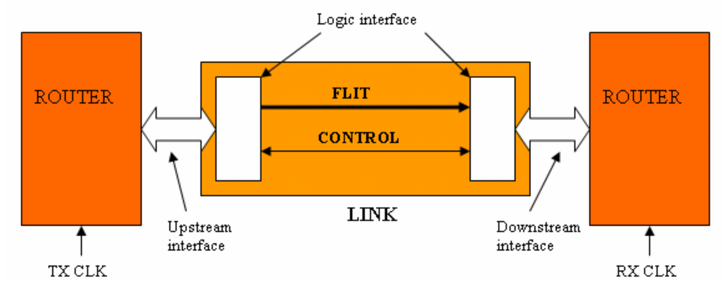


Figure A-10 Structure of the synchronous Spidergon STNoC Link.

The Spidergon STNoC link is synchronous, parallel and unidirectional. The link consists of two logic interfaces (upstream and downstream) and two set of wires (data and control). The link uses the credit based flow control mechanism to transmit data (flits). Signals sent on the control wires specify various properties of the transmitted flit (side-band information) such as its relative position in the packet (i.e. header, payload or final). The control wires also carry the signals necessary for credit based flow control (i.e. valid and credit). The credit based flit control does not allow packet dropping or retransmission, so each received flit must be accepted.

In 3D stacked MPSoCs, Spidergon STNoC can ensure intra-die and inter-die communication. In the next section, an MPSoC based on the Gaisler Aeroflex GRLIB IP library is extended with AMBA AXI support for Spidergon STNoC.

B.3 MPSoC Architecture

The MPSoC GRLIB template design comprises one or more LEON3 SPARC-V8 microprocessors, memory controllers, debug support unit, peripheral devices, and other IP blocks (e.g. cryptography, off-chip communication, etc). These system components are interconnected by a complex fabric that comprises an AMBA AHB multi-master bus, which connects CPUs, memory controllers, and an APB bus, which is accessed using an AHB / APB bus bridge and connects slow peripherals (e.g. UART, timers, etc).

The 3D MPSoC is a shared-memory system and it is extended with addressable components that are connected using Spidergon STNoC. Spidergon STNoC is AXI-compliant and, in order to access these modules, AHB / AXI bridges are connected to the AHB bus. The NoC has a custom topology and packets are routed using source-routing (i.e. the network interface indicates the path packets must take). In the system memory map, these modules occupy the address spaces *0xC0000000-0xC000FFFF* (MEM_1) and *0xC0010000-0xC001FFFF* (MEM_2), respectively.

The MPSoC is implemented on two dies with one die comprising the AHB-subsystem and the MEM_1 module with its network interface, and the second die comprising the MEM_2 module with its network interface and adjacent routers. The 3D Spidergon STNoC spans across two dies, ensuring intra-die access to MEM_1 and inter-die access to MEM_2. In Figure A-11, the 3D Spidergon STNoC-based MPSoC is represented.

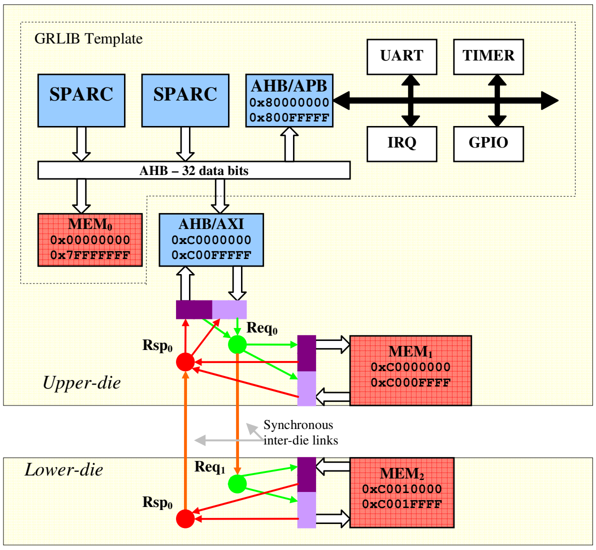


Figure A-11 Spidergon STNoC-based 3D MPSoC architecture

In order to avoid transaction deadlocks, there are two physically separated networks one for requests and one for responses. Each of the request / response networks has two routers (i.e. Req_0 and Req_1, Rsp_0 and Rsp_1) to which the *NI* of the MEM_1 and MEM_2 modules are connected. The lower-die components are connected to the rest of the system using two unidirectional synchronous Spidegon STNoC links. The request inter-die link has *92* data signals and *10* side-band information and control signals, while the response link has *64* data signals and *12* side-band information and control signals.

This extended system is LEON3-compliant and it is capable of running any operating system supported by the GRLIB IP library (e.g. eCos, RTEMS, Linux, etc.). The objective of this chapter is not to show how to design MPSoC, but to assess the impact of error resilience on its performance. Therefore, in terms of software, a series of simple applications that randomly access different memory locations (e.g. main memory, MEM_1 and MEM_2) are considered. In the following section, the error resilience improvements of the Spidergon STNoC are discussed.

B.4 Error Resilient 3D Spidergon STNoC Link

In the 3D integration setting, inter-die wire permanent and transient faults are a major source of yield and reliability loss. It is assumed that no fault tolerance mechanisms are necessary to address faults on intra-die components. Moreover, in-field TSV failure rates are considered to be very small, compared to the system expected lifetime. Therefore, the error resilience solution must cope with TSV permanent faults due to manufacturing defects and transient faults.

Once the target system and failure modes are specified, the *ERX* tool will assess the potential solutions. The NoC architecture is very simple (i.e. two routers for each sub-network) and only link-level solutions are considered. In Figure A-12, the selected error protection strategy is represented.

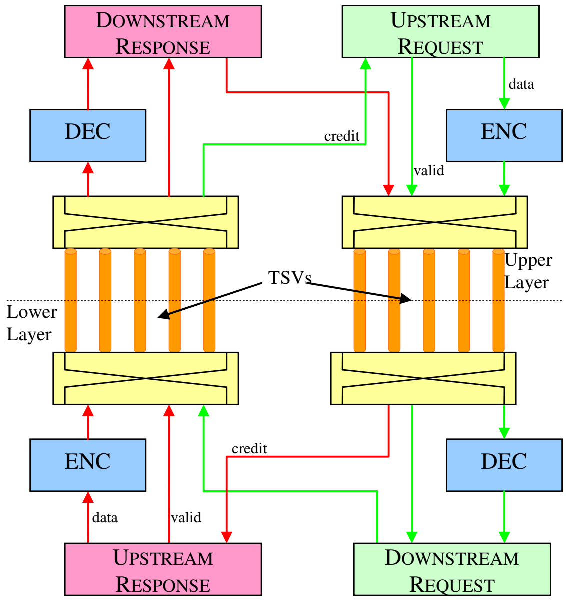


Figure A-12 Error resilience scheme for request and response links in the 3D MPSoC

Due to the system size, adding TSVs is not a real issue. Hence, the *Forward Error Correction* scheme is selected, as it offers good reliability, with lower costs (i.e. area / power overheads) than retransmission-based schemes. Data signals are protected using Hamming SEC codes, while critical signals are protected using *TMR*. Because Hamming decoding can be performed in a single clock cycle, without affecting the NoC timing, the request and response link decoders are implemented without the correction bypass stage. It is assumed that the single TSV wire error rates up to $\varepsilon_{wire}=10^{-5}$. Hence, as shown in Figure IV-3, the 99.9999% the reliability target (i.e. $\varepsilon_T=10^{-7}$) can be achieved both for request and response inter-die links.

Permanent TSV faults due to manufacturing defects are repaired using spare TSVs. Spares are allocated for each inter-die link such that regular TSVs (i.e. inter-die wires for data signals and error control bits) are repaired with a probability above the target yield $Y_T=99.99\%$. Note that Hamming codes can mask TSV permanent failures due to manufacturing and during system lifetime. However, the correction capabilities are limited to a single error and it is not possible to simultaneously have on a single link a transient and a permanent fault. Therefore, assuming a TSV defect rate of *0.01%*, one spare TSV is allocated for each link.

The area/power overheads of the error resilience strategy are less than *20%*, but the number of TSVs necessary for inter-die communication increases from *182* to *220* TSVs. The error protection has an impact on the network and system performance. Because a retiming stage is inserted at the decoder inputs, the latency of each link increases by one clock cycle. In the following section, a system-level assessment of this performance penalty is presented.

B.5 Performance Evaluation

In the MPSoC platform, the LEON3 microprocessors access the MEM_1 and MEM_2 modules using load/store operations. In the AHB-subsystem, CPU transaction requests are addressed to the AHB/AXI protocol bridge. When the AHB arbiter grants the master's request to the bridge, the AHB transaction is converted into an AXI transaction, which is packetized by the Spidergon STNoC network interfaces. These packets traverse the request network flit-by-flit and they are processed by the destination network interface, which sends the request to the MEM_1 or MEM_2 modules.

In terms of transaction run-time, adding error resilience on each inter-die link will increase the request / response message latency. Since only the inter-die links are protected, any performance degradation is due to the execution time of a load / store transaction having MEM_2 as target. When error resilience is added on the inter-die links, there is a penalty of two clock cycles. One cycle is due to the extra delay on the request inter-die link and the other cycle is due to the delay on the response inter-die link. In both cases, the extra delay does not depend on the error rate. This penalty occurs even in the fault-free case, as data encoding, transmission on PHY, and error detection, cannot be performed in a single cycle. A protocol run usually takes from tens to hundreds of clock cycles. Therefore, the performance penalty due to the two extra cycles is negligible even for a single transaction. Overall, the performance penalty depends on the number of transactions having MEM_2 as target. Hence, if fewer transactions have MEM_2 as target then the application performance overhead is smaller.

A series of applications that access different memory location have been considered. Experimental results have shown that the impact on the execution time is negligible. Therefore, it can be concluded that the major impact that the error resilience strategy has is not on the system performance, but on the number of TSVs.

Abstract: 3D technology promises energy-efficient heterogeneous integrated systems, which may open the way to thousands cores chips. Silicon dies containing processing elements are stacked and connected by vertical wires called Through-Silicon-Vias. In 3D chips, interconnecting an increasing number of processing elements requires a scalable high-performance interconnect solution: the 3D Network-on-Chip. Despite the advantages of 3D integration, testing, reliability and yield remain the major challenges for 3D NoC-based systems. In this thesis, the TSV interconnect test issue is addressed by an off-line Interconnect Built-In Self-Test (IBIST) strategy that detects both structural (i.e. opens, shorts) and parametric faults (i.e. delays and delay due to crosstalk). The IBIST circuitry implements a novel algorithm based on the aggressor-victim scenario and alleviates limitations of existing strategies. The proposed K^{th}-aggressor fault (KAF) model assumes that the aggressors of a victim TSV are neighboring wires within a distance given by the aggressor order K. Using this model, TSV interconnect tests of inter-die 3D NoC links may be performed for different aggressor order, reducing test times and circuitry complexity. In 3D NoCs, TSV permanent and transient faults can be mitigated at different abstraction levels. In this thesis, several error resilience schemes are proposed at data link and network levels. For transient faults, 3D NoC links can be protected using error correction codes (ECC) and retransmission schemes using error detection (Automatic Retransmission Query) and correction codes (i.e. Hybrid error correction and retransmission).For transients along a source-destination path, ECC codes can be implemented at network level (i.e. Network-level Forward Error Correction). Data link solutions also include TSV repair schemes for faults due to fabrication processes (i.e. TSV-Spare-and-Replace and Configurable Serial Links) and aging (i.e. Interconnect Built-In Self-Repair and Adaptive Serialization) defects. At network-level, the faulty inter-die links of 3D mesh NoCs are repaired by implementing a TSV fault-tolerant routing algorithm. Although single-level solutions can achieve the desired yield / reliability targets, error mitigation can be realized by a combination of approaches at several abstraction levels. To this end, multi-level error resilience strategies have been proposed. Experimental results show that there are cases where this multi-layer strategy pays-off both in terms of cost and performance. Unfortunately, one-fits-all solution does not exist, as each strategy has its advantages and limitations. For system designers, it is very difficult to assess early in the design stages the costs and the impact on performance of error resilience. Therefore, an error resilience exploration (ERX) methodology is proposed for 3D NoCs.

Keywords: 3D integration, Through-Silicon-Vias, 3D Networks-on-Chip, Error Resilience, Interconnect Built-In Self-Test

Résumé : Malgré les avantages de l'intégration 3D, le test, le rendement et la fiabilité des Through-Silicon-Vias (TSVs) restent parmi les plus grands défis pour les systèmes 3D à base de Réseaux-sur-Puce (Network-on-Chip - NoC). Dans cette thèse, une stratégie de test hors-ligne a été proposé pour les interconnections TSV des liens inter-die des NoCs 3D. Pour le TSV Interconnect Built-In Self-Test (TSV-IBIST) on propose une nouvelle stratégie pour générer des vecteurs de test qui permet la détection des fautes structuraux (open et short) et paramétriques (fautes de délaye). Des stratégies de correction des fautes transitoires et permanents sur les TSV sont aussi proposées aux plusieurs niveaux d'abstraction: data link et network. Au niveau data link, des techniques qui utilisent des codes de correction (ECC) et retransmission sont utilisées pour protéger les liens verticaux. Des codes de correction sont aussi utilisés pour la protection au niveau network. Les défauts de fabrication ou vieillissement des TSVs sont réparé au niveau data link avec des stratégies à base de redondance et sérialisation. Dans le réseau, les liens inter-die défaillante ne sont pas utilisables et un algorithme de routage tolérant aux fautes est proposé. On peut implémenter des techniques de tolérance aux fautes sur plusieurs niveaux. Les résultats ont montré qu'une stratégie multi-level atteint des très hauts niveaux de fiabilité avec un cout plus bas. Malheureusement, il n'y as pas une solution unique et chaque stratégie a ses avantages et limitations. C'est très difficile d'évaluer tôt dans le design flow les couts et l'impact sur la performance. Donc, une méthodologie d'exploration de la résilience aux fautes est proposée pour les NoC 3D mesh.

Mots-clef: Integration 3D, Through-Silicon-Vias, 3D Networks-on-Chip, Tolerance aux Fautes, Interconnect Built-In Self-Test